ALLER ABENTEUER ANFANG

Telematch Computer Praxis

TI99/4A Collection

TI-Vergangenheit: serielle und parallele Ports auf einer Karte

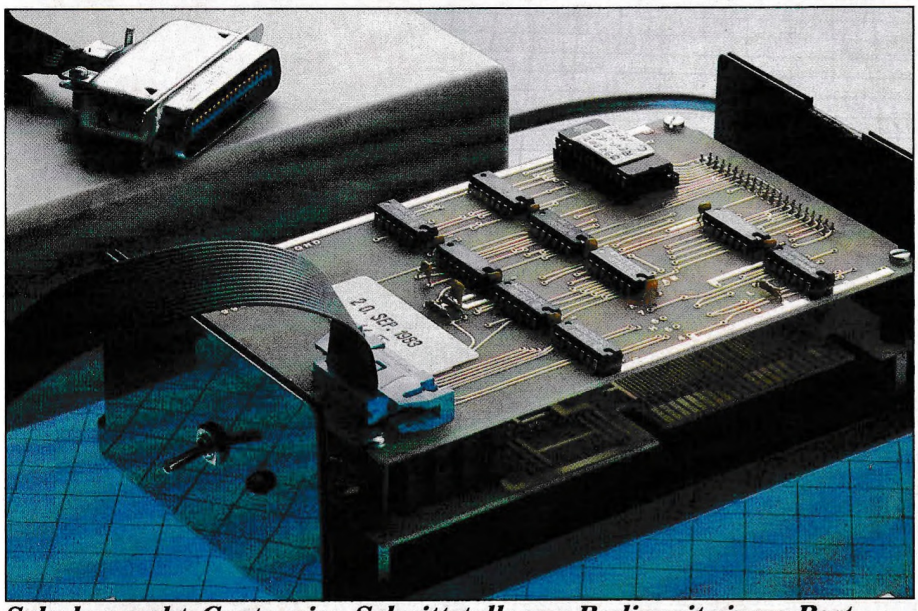
Schnittstellen: maßgeschneidert oder ausbaufähig

Sicher habt ihr auch schon mal daran gedacht, euren TI 99/4A Heimcomputer zu erweitern. COMPUTER Praxis hat diesmal zwei recht unterschiedliche Angebote unter die Lupe genommen: ein spezielles Drucker-Interface der Firma Radix und das V24-Interface der Firma Atronic.

Die Radix-Schnittstelle präsentiert sich als kleiner grauer Kasten, nur 10x5x20 cm (BxHxT) groß. Das einzige Bedienungselement ist ein an der Vorderseite angebrachter Netzschalter, der durch eine Leuchtdiode zur Anzeige der Betriebsbereitschaft ergänzt wird. Das Innere des Interface zeigt sich aufgeräumt. Die Elektronik ist auf zwei Platinen aufgebaut, die in Sandwich - Bauweise übereinander montiert sind, im Bild auf der rechten Seite zu erkennen.
Diesen Blick in's Innenleben habt ihr nicht (nur) der Neugierde der Tester bei COMPUTER Praxis zu verdanken, er wird auch jedem Anwender nicht erspart bleiben.

Maßgeschneidert: Das Radix-Interface

Denn bevor man das Interface in Betrieb nehmen kann, muß das mitgelieferte Druckerkabel angeschlossen werden. Dieses Kabel ist auf der einen Seite mit einem 36-poligen Stecker, auf der anderen mit einem 16-poligen DIL-Stecker ausgestattet. Der 36-polige Stecker entspricht nicht dem Centronics-Standard, und um den Anschluß an einen vorhandenen Drucker zu bewerkstelligen, wird man an den Fachhandel verwiesen. In der Praxis wird man wohl selber zum Lötkolben greifen müssen. Immerhin ist in der Bedienungsanleitung die Pinbelegung abgedruckt. Aber den zum DIL-Stecker passenden Sockel sucht man zunächst vergeblich, er wird erst nach Öffnen des Gehäuses zugänglich. Diese Lösung wirkt nicht gerade professionell, doch immerhin ist an der Rückseite des Gehäuses eine Aussparung angebracht, so daß das Kabel ordentlich nach draußen geführt werden kann.
Nachdem das Gehäuse wieder geschlossen wurde, fehlt zur Inbe-

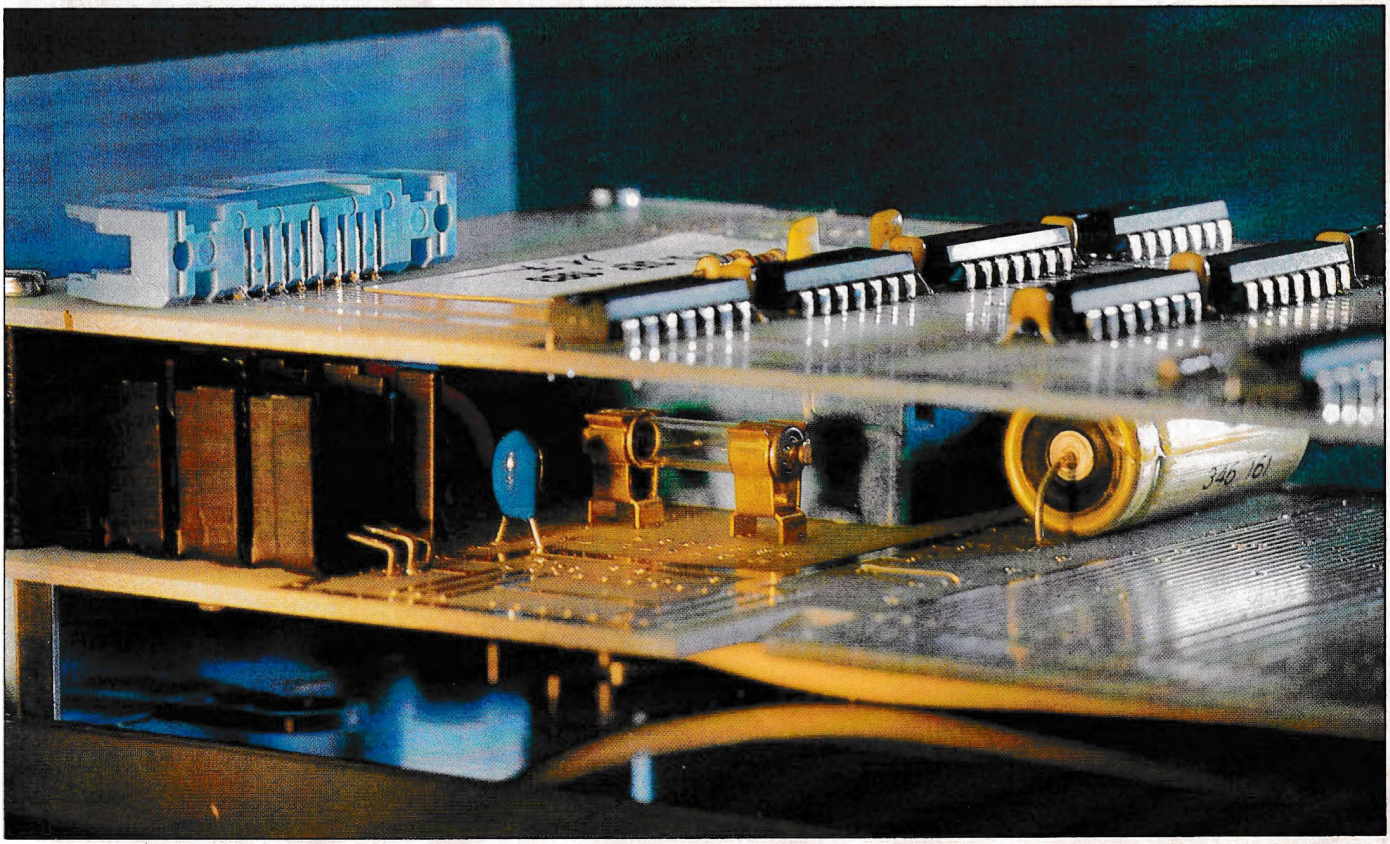


Sehr kompakt: Centronics-Schnittstelle von Radix mit einem Port — das Netzteil wird von der oberen Platine verdeckt.

triebnahme nur noch die Verbindung mit dem Rechner. Das Interface verfügt dazu auf der linken Seite über einen an die Platine gelöteten Stecker zum Anschluß an den Systembus des TI 99/4A und auf der rechten Seite über eine Steckleiste, an welcher der Systembus wieder herausgeführt wird. Die entsprechende Steckleiste am TI-Rechner befindet sich an der rechten Seite unter einer Klappe. Um einen festen Halt des angeschlossenen Steckers zu gewährleisten, ist diese Steckleiste am TI zusätzlich mit Klemmblechen versehen. Und genau diese bereiten auch das nächste Problem, denn der Stecker des Radix-Interface will sich hier nur unwillig einführen lassen. Folgt man der Bedienungsanleitung, so sollen diese Bleche vorsichtig auseinander gebogen werden, aber natürlich auch nicht zu weit, damit sie ihre Aufgabe noch erfüllen können. Wir meinen, das Interface sollte mit einem anderen Stecker versehen werden; denn es ist eine arge Fummelei, die Bleche zu verbiegen, womöglich gleichzeitig das Interface anzustecken, ohne etwas zu beschädigen.

Doch nachdem auch diese Hürde genommen war, zeigte sich das

Foto unten: Nachteil der Sandwich-Anordnung — die Sicherung auf der unteren Platine ist schwer zugänglich. Um sie zu wechseln, muß die obere abgenommen werden.



Radix Interface gut aufgelegt, und das Drucken funktionierte problemlos. Für den Fall der Fälle steht die ausführliche und gut gegliederte Bedienungsanleitung sogar mit einer kurzen Testroutine zur Seite.

Ausbaufähig: das Atronic-Peripherie-System

Mit dem Anschluß eines Druckers sind die Möglichkeiten des Radix-Interface allerdings erschöpft. Ganz anders das viel umfangreicher angelegte System von Atronic. Sieht man sich das Angebot an, glaubt man sich in eine andere Welt versetzt. Alle Geräte haben Alu-Gehäuse mit den Abmessungen 15 x 8 x 25 cm. Sie werden an den System-Bus des TI 99/4A angeschlossen, der durch alle Atronic-Geräte hindurchgeführt wird, so daß eine umfangreiche Peripherie angeschlossen werden kann. Alle Geräte haben, soweit erforderlich, ein eingebautes Netzteil. Erhältlich sind bis jetzt:

- ▷ V24-Interface mit 2 Ports
- ▷ 32 KB RAM Erweiterung
- ▷ 32 KB CMOS RAM,
  a) flüchtig
  b) nicht flüchtig

Weitere Geräte, die in der Entwicklung sind:

- ▷ Disk-Controller
- ▷ Modem
- ▷ EPROM-Brenner
- ▷ Bus-Kabel
- ▷ 64 KB RAM

Universell: die RS 232-Karte

Wir hatten das V24-Interface im Test, das zwei serielle Ports zur Verfügung stellt. Kabel, um das Interface mit dem Peripheriegerät zu verbinden, werden vom Fachhandel angeboten. Der Zugang zu den Anschlüssen ist ohne Öffnen der Box möglich. Auch das Interface hat ein Alu-Gehäuse und macht einen soliden Eindruck. Es kann ohne Schwierigkeiten sicher an den Bus des TI 99/4A gesteckt werden; Probleme mit den oben beschriebenen Klemmblechen traten nicht auf. Um ganz sicherzugehen, haben wir das Interface an mehrere Terminals angeschlossen.

Da die Firma Atronic bald ein Kabel herausbringt, wird auch die Möglichkeit gegeben, die doch recht großen Zusatzgeräte beiseite zu stellen. Im Gespräch ist auch ein kurzes Kabel, um die Geräte stapeln zu können.

Die Benutzung des V24-Interface ist eine wahre Freude. Zunächst glänzt es durch seine genaue Betriebsanleitung mit Anschlußanweisungen, einer Liste der Anschlußbelegung, Beispiel-Routinen und einer Erklärung möglicher Fehlercodes. Ganz hervorragend sind die Möglichkeiten zur Einstellung verschiedener Übertragungsformate: Alle Optionen sind durch Software einstellbar, das mühsame Einstellen kleiner Schalter, wie es sonst häufig zu finden ist, entfällt. In Tabelle 1 sind die möglichen Funktionen zusammengestellt.

Tabelle 1: **Einstellbare Optionen des Atronic V-24-Interface**

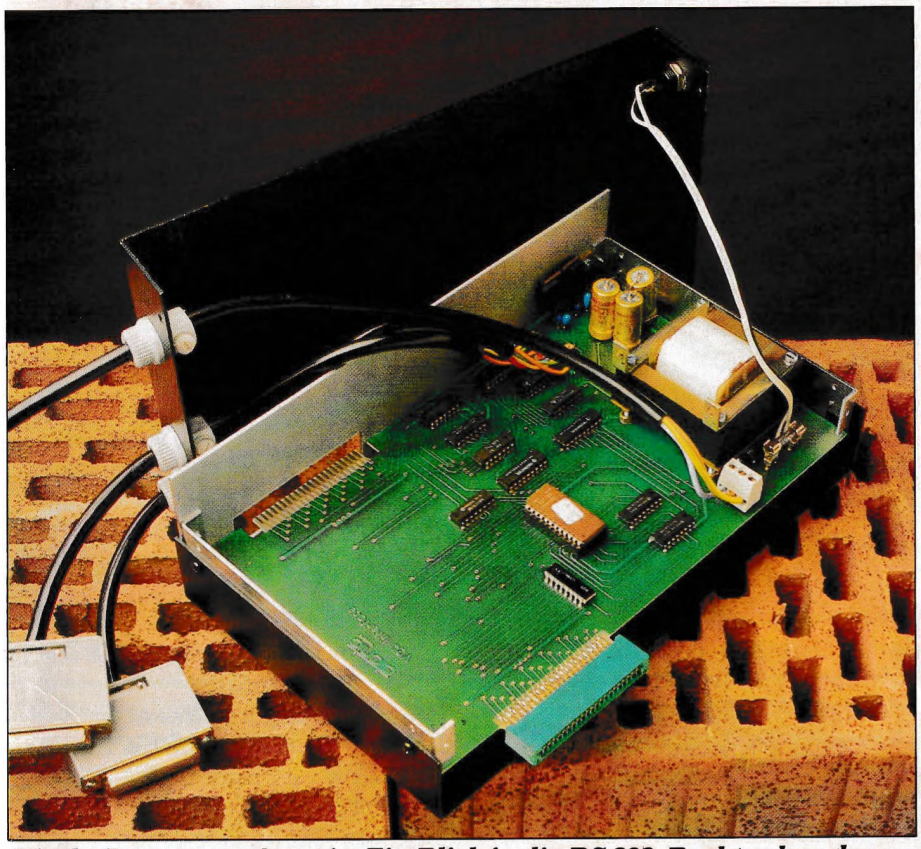
Option	Funktion	Mögliche Werte (Standardwert)
.BA=	Baud-Rate	110, (300), 600, 1200, 2400, 4800, 9600
.DA=	Anzahl Daten-Bits	(7), 8
.PA=	Art des Parity-Bits (odd, even, none)	(O), E, N
.TW	2 Stop-Bits	(1 Stop-Bit)
.NU		
.CH	Parity-Prüfung einschalten	(keine Prüfung)
.EC	Echo ausschalten	(Echo ein)
.CR	Wagenrücklauf aus	
.LF	Zeilenvorschub und Wagenrücklauf aus	

Bei den Optionen .BA=, .DA=, .PA= muß nach dem Gleichheitszeichen einer der rechts angegebenen Werte folgen. Die anderen Optionen schalten eine bestimmte Funktion ein, anderenfalls gilt die in der rechten Spalte aufgeführte Funktionsweise.
Benutzt man die Schnittstelle im Zusammenhang mit OLD- oder SAVE-Befehlen, können nur die Optionen .BA= und .TW benutzt werden.
Durch diesen Befehl wird folgendes Übertragungsformat gewählt: Baud-Rate 9600, 7 Datenbits, 1 Stop-Bit, keine Nullen, Prüfung auf ungerades (odd) Parity-Bit, kein Echo.

Liste 1: **Test-Programm (Radix-Version)**

```
100 OPEN #2:"PRT!!, OUTPUT   Eröffnen einer Datei zum Drucken
110 FOR I = 33 TO 126        I ist Index in ASCII-Tabelle
120 PRINT #2:CHR$(I)         Drucken eines Zeichens
130 NEXT I                   Schleife für alle
                             darstellbaren Zeichen
140 PRINT #2                 Einfügen einer Leerzeile
150 CLOSE #2                 Schließen der Datei
```

ACHTUNG: Für die Atronic-Schnittstelle lautet der erste Befehl:
100 OPEN #2:"RS232",OUTPUT



Block-System von Atronic: Ein Blick in die RS 232. Rechts oben das Netzteil, links die 2 Ports, in der Mitte das steuernde EPROM.

Problemlos: Drucken mit dem TI 99/4A

Der Betrieb eines Druckers ist mit der Radix-Schnittstelle und dem V24-Interface praktisch gleich. Um ein Programm auszudrucken genügt der Befehl

LIST "PRT" (Radix) bzw.

LIST "RS 232" (Atronic)

Als zusätzlicher Parameter kann ein Zeilenbereich angegeben werden, wenn man nur einen Programmteil ausgeben möchte. So listet der Befehl

LIST "PRT":210-440 (Radix) bzw.

LIST "RS 232":210-440 (Atronic)

nur die Programmzeilen 210 bis 440.

In der Liste 1 finden Sie ein kurzes Programm, das alle darstellbaren ASCII-Zeichen auf einem angeschlossenen Drucker ausgibt. Wenn Sie in diesem Programm den Befehl in Zeile 150 durch GO-TO 110 ersetzen, haben Sie eine einfache Dauertest-Routine. Der Drucker arbeitet dann solange, bis das Programm durch FCTN CLEAR unterbrochen wird.

Der im Programm verwendete OPEN-Befehl verlangt eine kurze Erklärung. „OPEN" eröffnet eine Datei und teilt dem Computer dadurch mit, wohin die vom Programm erzeugten Daten geschrieben werden sollen. Die Parameter dieses Befehls bestimmen, ob eine parallele, serielle Schnittstelle oder vielleicht eine Floppy-Disk angeschlossen sind. In unserem Beispiel

100 OPEN #2:"PRT", OUTPUT

bedeutet #2 eine Dateinummer, "PRT" bzw. "RS 232" bestimmen die Art der Schnittstelle und OUTPUT zeigt an, daß Daten ausgegeben werden sollen. Weitere Parameter wären möglich, aber wenn man sie wie hier nicht angibt, wählt der Rechner selbständig sinnvolle Standardwerte.

Als Fazit kann man sagen, daß sich das Problem eines Druckeranschlusses mit beiden getesteten Geräten lösen läßt. Entscheidungskriterium sollte hier sein, wie weit man sein System ausbauen möchte.

Die Qual der Wahl

Das Radix-Interface bietet zu einem Preis von ca. 300 Mark eine gerade noch preiswert zu nennende Alternative. Wegen des durchgeschleiften Systembusses ist der Anschluß weiterer Interfaces immerhin möglich.

Wer sich für das V24-Interface von Atronic entscheidet, steigt preislich höher ein (ca. DM 390,-- mit einem, DM 440,-- mit zwei Anschlüssen), hat jedoch außer dem durchgeschleiften Bus nicht nur einen, sondern gleich zwei V24-Anschlüsse. Dazu kommt, daß bei beiden Anschlüssen getrennt voneinander die für das Peripherie-Gerät nötigen Optionen (Baud-Rate, Data-Bits etc.) eingestellt werden können. Man sollte auch nicht vergessen, daß die Geräte der Atronic-Systems aufeinander abgestimmt sind. So gesehen dürfte es sicherlich die kostengünstigere Lösung sein.

Ganz bewußt ist darauf verzichtet worden, das Radix- und das Atronic-Interface mit dem Expansion-System von TI zu vergleichen. Der Wert der beiden Geräte als Ersatz für die von TI nicht mehr produzierten Erweiterungen liegt auf der Hand. Herr Vernunft von der Firma Atronic erklärte uns, daß die Firma sich auf TI-Peripherien spezialisiert hat und alle Geräte mit Texas Instruments voll kompatibel sind, z. T. baugleich, z. T. überarbeitet und neu konzipiert wurden. ■

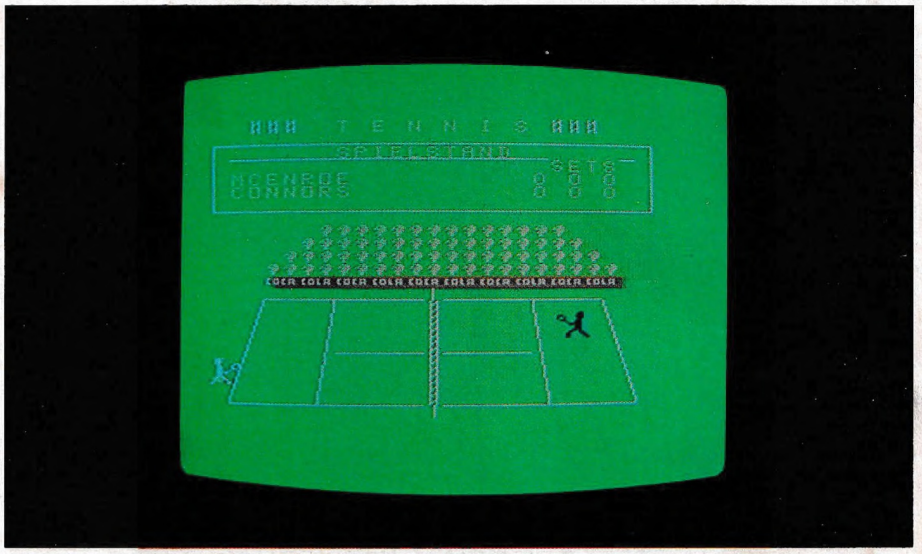
Das Atronic V24-Interface gibt es bei: Atronic Bunke & Vernunft oHG, Postfach 70 01 24, 2000 Hamburg 70, Tel: 668 22 56.

Die Radix Drucker-Schnittstelle bei: Radix Büro-Technik, Bornstraße 4, 2000 Hamburg 13, Tel: 410 25 47.

3-D-TENNIS

Wimbledon im Computer

Jahre sind vergangen, seit der Urgroßvater der Videospiele, ein Programm namens „Pong", die Kaufhäuser bevölkerte. Noch immer geistert die Idee in den Köpfen der Programmierer umher, das Tennisspiel auf dem Bildschirm möglichst realistisch zu gestalten. Das zeigen jüngste Veröffentlichungen führender Software-Anbieter wie ATARI und MATTEL.
Auch das vorliegende Spiel ist ein Tennisprogramm, das gegenüber „Pong" an einer Dimension und zahlreichen Raffinessen gewonnen hat. Das Programm läuft auf dem TI99/4A und EX-BASIC.



Das Spielfeld:
Hat man das Programm geladen und gestartet, erscheint auf dem Bildschirm ein 3-D Tennisfeld, eine dazugehörige Zuschauertribüne und eine Anzeigetafel, in die man die Namen der Spieler eintragen kann.

Die Steuerung:
Die Spieler werden mit zwei Joysticks über das Feld bewegt. Will ein Spieler den Ball schlagen, muß er im entsprechenden Augenblick die rote Taste drücken. Die Flugbahn des Balles kann er dadurch beeinflussen, daß er den Stick in diesem Moment in die Richtung drückt, in die er den Ball lenken möchte.

Die Zählweise:
Gezählt wird, wie es der Internationale Tennisverband vorschreibt. Gespielt werden zwei Gewinnsätze. Ein Tiebreak ist im Programm vorgesehen und wird bei entsprechendem Spielstand ausgeführt.

Die Regeln:
Der Ball muß gespielt sein, bevor er ein zweites Mal aufprallt. Der Ball darf volley gespielt werden, er darf nicht ins Aus oder ins Netz geschlagen werden.

Zusätzliche Möglichkeiten:
Nach jedem Ballwechsel kann durch Drücken einer Taste eine Zeitlupenwiederholung eingespielt werden. Hat einer der Spieler bereits die beiden ersten Sätze gewonnen, kann dennoch auf Wunsch der dritte gespielt werden.

Die Initialisierung:
Dieser Programmteil eignet sich hervorragend zum Experimentieren, da man, wenn man Zeilennummer und Befehl kennt, mit wenig Aufwand viel Veränderung auf dem Bildschirm erreichen kann. Deshalb sind die Zeilennummern mit entsprechenden Befehlen zur Veränderung der Farben und den in Erscheinung tretenden Zeichen gelistet.

Das Hauptprogramm:
Vom Hauptprogramm werden nacheinander die Bewegungsroutinen der einzelnen Spieler aufgerufen. Danach wird jeweils die Schlagroutine des Spielers, der als nächster schlagen muß, ausgeführt. In Zeile 510 wird mit CALL POSITION (*3,PY,PX) die Position des Balles ermittelt und überprüft, ob der Ball ein erstes (IND=0) oder ein zweites Mal (IND=1) aufgesprungen ist.

Die Bewegungsroutinen:
In den Routinen „BEWEGUNG SPIELER 1" und „BEWEGUNG SPIELER 2" werden der Joystick und der Sprite des Spielers entsprechend bewegt. Diese Routine kann auch für andere Spiele übernommen werden. Man muß lediglich vorher mit CALL SPRITE (*1,...,...) einen Sprite definieren.
Bei den Schlagroutinen in Zeile 610 und 680 wird zunächst abgefragt, ob der rote Knopf am Joystick gedrückt wurde. Wurde der Knopf nicht gedrückt, springt der Programmablauf mit RETURN wieder zurück ins Hauptprogramm. Wurde jedoch der Knopf gedrückt, wird mit CALL COINC(ALL) abgefragt, ob der Ball getroffen wurde. Ist das der Fall, so erfolgt eine Richtungsänderung des Balles gemäß der Auslenkung des Joysticks in Zeile 620.

Das Zählwerk:
Dieses Unterprogramm enthält die Regeln der komplizierten

Zeile	100— 480	Initialisierung
Zeile	490— 550	Steuer- und Hauptprogramm
Zeile	560— 910	Unterprogramm zur Steuerung der Spieler
Zeile	920—1270	Unterprogramm für das Tenniszählwerk
Zeile	1280—1310	Unterprogramm für das Bewegen der Zuschauerköpfe
Zeile	1320—1440	Unterprogramm der Zeitlupe

Die Farben:		
Hintergrund-Bildschirm	330	CALL SCREEN (X)
Hintergrund-Feld	330	CALL COLOR (1,X,X)
Schrift und Zahlen	340	CALL COLOR (3,X,X)
		CALL COLOR (I,X,X)
Feldumrandung	330	CALL COLOR (2,X,X)
		CALL COLOR (4,X,X)
		CALL COLOR (8,X,X)
Spieler links	470	CALL SPRITE (#2,121,X,...)
Spieler rechts	470	CALL SPRITE (#1,101,X,...)
Ball	470	CALL SPRITE (#3,132,X,...)
Die Zeichen:		
Spieler, stehend rechts	400	S1$="XXXXXXXX..."
Spieler, stehend links	410	S1$="XXXXXXXX..."
Spieler, schlagend rechts	420	S3$="XXXXXXXX..."
Spieler, schlagend links	430	S4$="XXXXXXXX..."
Köpfe der Zuschauer	100	CALL CHAR (140,"XXXX...")
		CALL CHAR (141,"XXXX...")
	130	CALL CHAR (142,"XXXX...")
Inschrift Bandenwerbung	110	CALL CHAR (112,"XXXX...")
		CALL CHAR (113,"XXXX...")
		CALL CHAR (114,"XXXX...")
Ball	450	CALL CHAR (132,"XXXX...")
Netz	250	CALL CHAR (89,"XXXX...")

Zählweise des Internationalen Tennisverbandes und wird nach Abschluß eines jeden Ballwechsels aufgerufen.
Interessant dürfte dabei die Zählweise des Computers sein, denn bekanntlich zählt man beim Tennis „0,15,30,40,Spiel" und es tauchen Begriffe wie „Einstand" und „Vorteil" auf. Da es jedoch einfacher ist, den Computer „1,2,3,4,..." zählen zu lassen, sind die Werte 0,15,30,40 sowie die Begriffe „Spiel", „Einstand" und „Vorteil" in die Variablen P$(1), P$(2), P$(3) usw. eingelesen. Auf diese Weise braucht nur die Zahl in der Klammer um eins erhöht zu werden, um den aktuellen Spielstand zu erhalten. Diese Technik wird häufiger verwendet, um unregelmäßige Wertzunahmen in Abhängigkeit von regelmäßigen Wertzunahmen zu setzen.

Die Zeitlupe:
In den Unterprogrammen SCHLAG SPIELER 1 + 2 kommen eine Menge Wertzuweisungen zu Variablen vor, die scheinbar keinen Nutzen erfüllen. Diese Variablen speichern die Positionen der Spieler und die Position sowie die Bewegung des Balles für die hier ausgeführte Zeitlupe.
Das Prinzip ist bei dieser Zeitlupe ähnlich wie beim Trickfilm: die den Schlagroutinen entnommenen Momentaufnahmen (abgespeichert in den Variablen) werden hintereinander mit CALL LOCATE auf den Bildschirm gebracht und ergeben so ein Bild vom Gesamtgeschehen. Man könnte natürlich die etwas ruckartige Darstellung der Zeitlupe verfeinern, indem man im Hauptprogramm und nicht im Schlagprogramm die Variablenzuweisung durchführte. Das würde jedoch das Programm über die Maßen verlangsamen.
Ich wünsche viel Spaß mit „TENNIS" und hoffe, euch Anregungen zum Verändern dieses Programms gegeben zu haben.

T E N N I S

```
100 CALL CLEAR :: CALL CHAR(140,"001C2E7E1E3C1818")::
    CALL CHAR(141,"0038747E783C1818")::
    CALL COLOR(14,12,1):: DIM SZ1(40),SS1(40),SZ2(40)
110 DIM SS2(40),BRS(40),BRZ(40),BZ(40)::
    CALL CHAR(112,"00006E4A4A6E")::
    CALL CHAR(113,"0000DC949CD4")::
    CALL CHAR(114,"00009C949CD4")
120 DIM BS(40)
130 CALL CHAR(142,"00387C547C3810")::
    CALL COLOR(11,16,7):: CALL SCREEN(2)
140 DATA 84,69,78,78,73,83
150 CALL MAGNIFY(2):: FOR I=1 TO 6 :: READ A ::
    CALL SPRITE(#I,A,INT(RND*10)+3,15+I*20,30+I*30)::
    NEXT I
160 CALL COLOR(1,2,2):: CALL COLOR(4,2,2)::
    CALL COLOR(8,2,2):: CALL COLOR(2,2,2)::
    CALL SCREEN(2)
170 DATA 4,0,8,0,1,0,2,0,0,4,0,8,0,1,0,2
180 DATA 8,0,4,0,2,0,1,0,0,8,0,4,0,2,0,1
190 FOR I=40 TO 43 :: FOR J=1 TO 2 :: READ A,B ::
    FOR K=1 TO 4 :: A$=A$&STR$(A)&STR$(B):: NEXT K ::
    NEXT J :: CALL CHAR(I,A$):: A$=""
200 NEXT I
210 FOR I=44 TO 47 :: FOR J=1 TO 2 :: READ B,A ::
    FOR K=1 TO 4 :: A$=A$&STR$(B)&STR$(A):: NEXT K ::
    NEXT J
220 CALL CHAR(I,A$):: A$="" :: NEXT I
230 CALL CHAR(94,"FF")::
    CALL CHAR(95,"0000000000000FF")
240 RESTORE 180 :: FOR I=56 TO 63 :: READ A,B ::
    FOR J=1 TO 8 :: A$=A$&STR$(A)&STR$(B):: NEXT J ::
    CALL CHAR(I,A$):: A$="" :: NEXT I
250 CALL CHAR(89,"242C342424C3424")
260 PRINT "   YYY  T E N N I S YYY   "
270 PRINT " {25*CHR$(95)}   8{6 SPACE}SPIELSTAND
    {7SPACE}?   8{18*CHR$(94)}SETS{CHR$(94)}?   8
    {7*CHR$(95)}{10 SPACE ]0 0 0 ?"
280 PRINT " 8{7*CHR$(95)}{10 SPACE}0 0 0 ?   8
    {23 SPACE}?   {25*CHR$(94)}{34 SPACE}
290 PRINT ,,,,,,
300 PRINT "{4 SPACE}{9*CHR$(95)};{9*CHR$(95)}
    {8 SPACE}+{3 SPACE}?{5 SPACE}Y{5 SPACE}8   ,
    {7 SPACE}*   >5 SPACE}Y{5 SPACE}9   -{7 SPACE}
    )   ={5 SPACE}Y{5 SPACE}:{3 SPACE}."
310 PRINT "   (   <{5 SPACE}Y{5 SPACE};   /{6 SPACE}+
    {4 SPACE};{5*CHR$(94)}Y{5*CHR$(94)}<{4 SPACE},
    {5 SPACE}*{4 SPACE}:{5 SPACE}Y{5 SPACE}={4 SPACE}
    -{5 SPACE}){4 SPACE}9{5 SPACE}Y{5 SPACE}>
    {4 SPACE}."
320 PRINT "  ({4 SPACE}8{5 SPACE}Y{5 SPACE}?{4 SPACE}
    /{5 SPACE}{11*CHR$(94)}<{11*CHR$(94)}   "
```

```
330 CALL DELSPRITE(ALL):: CALL COLOR(2,16,13)::
    CALL COLOR(4,16,13):: CALL COLOR(8,16,13)::
    CALL COLOR(1,16,13):: CALL SCREEN(13)
340 FOR BU=5 TO 7 :: CALL COLOR(BU,12,1):: NEXT BU ::
    CALL COLOR(3,12,1):: ZUS=142 :: GOSUB 1280
350 DISPLAY AT(24,3):"WIE HEISSEN DIE SPIELER " ::
    ACCEPT AT(5,3)VALIDATE(UALPHA)SIZE(-7):N2$ ::
    ACCEPT AT(6,3)VALIDATE(UALPHA)SIZE(-7):N1$
360 G(1)=191 :: G(2)=157 :: G(3)=124 :: G(4)=111 ::
    G(5)=78 :: G(6)=44 :: G(7)=8 :: G(0)=234
370 P$(1)="0" :: P$(2)="15" :: P$(3)="30" ::
    P$(4)="40" :: P$(5)="EINSTAND" ::
    P$(6)="VORTEIL" :: P$(7)="SPIEL"
380 P(1),P(2)=1 :: SP(1)=0 :: SP(2)=0 :: S=1
390 AUF=2 :: T=0 :: E$=""
400 S1$="0000E090907008070200010306OC0C0C7070F07060E0
    E0E0E0E0E0A030180C06"
410 S2$="0F0E0F0E060707070707070507C081030800000070909
    0E10E0400080C0603030"
420 S3$="0C12120C080B0F090103030306OC1800000000C0C0E0
    E0F8C0C0603018C06000"
430 S4$="0000000303070710F0303060C8103060030484830100D0
    F09080C0C0C060308100"
440 CALL CHAR(124,S4$):: CALL CHAR(104,S3$)::
    ZUS=140 :: GOSUB 1280
450 CALL CHAR(132,"0000000000000030707030000000000000
    0000000000C0E0E0C0000000000000")
460 CALL MAGNIFY(3):: CALL CHAR(100,S1$)::
    CALL CHAR(120,S2$)
470 CALL SPRITE(#1,101,2,150,200,0,0)::
    CALL SPRITE(#2,121,16,120,50,0,0)::
    CALL SPRITE(#3,132,12,120,120,0,0)
480 DISPLAY AT(24,1):" "
490 GOSUB 830
500 GOSUB 560 :: GOSUB 580 ::
    IF BR=-1 THEN GOSUB 670 ELSE GOSUB 600
510 CALL POSITION(#3,PY,PX)::
    IF ABS(PX-G(GR))>15 THEN 500
520 IF IND=1 THEN IND=0 ::
    DISPLAY AT(24,6):"2 MAL AUFGESPRUNGEN " ::
    GOTO 1160 ELSE IND=1 :: GR=GR-BR ::
    CALL SOUND(50,-7,3)
530 CALL MOTION(#3,HY/2,7*BR)
540 IF PY<105 OR PY>167 THEN BR=-BR ::
    DISPLAY AT(24,12):"A U S" :: GOTO 1160 ELSE 500
550 GOTO 500
560 ! BEWEGUNG SPIELER 1
570 CALL JOYST(1,X,Y):: CALL MOTION(#1,2*-Y,2*X)::
    RETURN
580 ! BEWEGUNG SPIELER 2
590 CALL JOYST(2,X,Y):: CALL MOTION(#2,2*-Y,2*X)::
    RETURN
600 ! SCHLAG SPIELER 1
610 CALL KEY(1,A,B)::
    IF A<>18 THEN RETURN ELSE CALL PATTERN(#1,105)::
    CALL COINC(ALL,C)::
    IF C=0 THEN CALL PATTERN(#1,101):: RETURN
620 CALL JOYST(1,X,Y)::
    CALL MOTION(#3,-Y,INT(-16+X*1.5))):: GR=X/-4+5 ::
    CALL SOUND(50,-5,5):: CALL PATTERN(#1,101):: BR=-1
630 IF IND=0 THEN GR=GR-1 ::
    IF GR<4 THEN DISPLAY AT(24,10):"N E T Z" ::
    BR=-BR :: GOTO 1160
640 WE=WE+1 :: CALL POSITION(#1,SZ1(WE),SS1(WE),
    #2,SZ2(WE),SS2(WE),#3,BZ(WE),BS( WE))::
    BRZ(WE)=-Y/2
650 BRS(WE)=INT((-16+X*1.5)/2)
660 IND=0 :: HY=-Y :: RETURN
670 ! SCHLAG SPIELER 2
680 CALL KEY(2,A,B)::
    IF A<>18 THEN RETURN ELSE CALL PATTERN(#2,125)::
    CALL COINC(ALL,C)::
    IF C=0 THEN CALL PATTERN(#2,121):: RETURN
690 CALL JOYST(2,X,Y)::
    CALL MOTION(#3,-Y,INT(16+X*1.5))):: GR=X/-4+2 ::
    CALL SOUND(50,-6,5):: CALL PATTERN(#2,121):: BR=1
700 IF IND=0 THEN GR=GR+1 ::
    IF GR>3 THEN DISPLAY AT(24,10):"N E T Z" ::
    BR=-BR :: GOTO 1160
710 WE=WE+1 :: CALL POSITION(#1,SZ1(WE),SS1(WE),
    #2,SZ2(WE),SS2(WE),#3,BZ(WE),BS(WE))::
    BRZ(WE)=-Y/2
720 BRS(WE)=-INT((-16+X*1.5)/2)
730 IND=0 :: HY=-Y :: RETURN
740 ! AUFSCHLAG SPIELER 1
750 CALL MOTION(#1,0,0,#2,0,0,#3,0,0)::
    CALL LOCATE(#1,160,217,#3,161,216,#2,120,40)::
    CALL PATTERN(#1,101)
760 CALL KEY(1,K,U):: GOSUB 580 ::
    IF U<>0 AND K<>18 THEN GOSUB 1320 :: GOTO 750
    ELSE IF K<>18 THEN 760 ELSE BR=-1 :: ZUS=141 ::
    GOSUB 1280
770 DISPLAY AT(24,1):" "
780 CALL JOYST(1,X,Y):: GR=4+ABS(X/4):: ZUS=140 ::
    CALL PATTERN(#1,105)
790 GOSUB 1280 :: CALL SOUND(50,-5,5)::
    CALL MOTION(#3,INT(-5-Y/3),-16-ABS(X))::
    CALL PATTERN(#1,101):: HY=-4
800 WE=1 :: CALL POSITION(#1,SZ1(WE),SS1(WE),
    #2,SZ2(WE),SS2(WE),#3,BZ(WE),BS(WE))):: BRZ(WE)=-3
810 BRS(WE)=-12
820 RETURN
830 ! AUFSCHLAG SPIELER 2
840 CALL MOTION(#1,0,0,#2,0,0,#3,0,0)::
    CALL LOCATE(#2,150,22,#3,155,20,#1,120,180)::
    CALL PATTERN(#2,121)
850 CALL KEY(2,K,U):: GOSUB 560 ::
    IF U<>0 AND K<>18 THEN GOSUB 1320 :: GOTO 840
    ELSE IF K<>18 THEN 850 ELSE BR=1 :: ZUS=140 ::
    GOSUB 1280
860 DISPLAY AT(24,1):" "
870 CALL JOYST(2,X,Y):: GR=3-ABS(X/4):: ZUS=141 ::
    CALL PATTERN(#2,125)
880 GOSUB 1280 ::
    CALL MOTION(#3,INT(-5-Y/3),16+ABS(X))::
    CALL SOUND(50,-6,5):: CALL PATTERN(#2,121):: HY=-4
890 WE=1 :: CALL POSITION(#1,SZ1(WE),SS1(WE),
    #2,SZ2(WE),SS2(WE),#3,BZ(WE),BS(WE))):: BRZ(WE)=-4
900 BRS(WE)=11
910 RETURN
920 !       ZAEHLWERK
930 IF T=1 THEN 1120
940 IF BR=-1 THEN P(1)=P(1)+1 ELSE P(2)=P(2)+1
950 IF P(1)=4 AND P(2)=4 THEN 980
960 IF P(1)<5 AND P(2)<5
    THEN DISPLAY AT(5,11):P$(P(2));::
    DISPLAY AT(6,11):P$(P(1));:: RETURN
970 IF ABS(P(1)-P(2))>1 THEN
    IF P(1)>P(2)THEN SP(1)=SP(1)+1 :: GOTO 1010
    ELSE SP(2)=SP(2)+1 :: GOTO 1010
980 IF P(1)=P(2)THEN DISPLAY AT(5,11):"EINSTAND";::
    DISPLAY AT(6,11):"EINSTAND"; :: RETURN
990 IF P(1)<P(2)THEN DISPLAY AT(5,11):"VORTEIL ";::
    DISPLAY AT(6,11):"{7*CHR$(94)}"; :: RETURN
```

```
1000 DISPLAY AT(5,11):"{7*CHR$(94)}"; ::
     DISPLAY AT(6,11):"VORTEIL "; :: RETURN
1010 IF T=1 THEN E$="ENDE" :: RETURN
     ELSE DISPLAY AT(24,6):"AUFSCHLAGWECHSEL " ::
     IF AUF=1 THEN AUF=2 ELSE AUF=1
1020 DISPLAY AT(5,11):"0       ";::
     DISPLAY AT(6,11):"0       ";::
     DISPLAY AT(5,S*2+17):SP(2);::
     DISPLAY AT(6,S*2+17):SP(1);
1030 IF SP(1)<6 AND SP(2)<6 THEN P(1)=1 :: P(2)=1 ::
     RETURN
1040 F=ABS(SP(1)-SP(2))
1050 IF F=0 AND SP(1)=6 THEN 1110
1060 IF F=1 AND(SP(1)=5 OR SP(2)=5)THEN P(1),P(2)=1 ::
     RETURN
1070 IF F>1 THEN S=S+1 :: P(1)=1 :: P(2)=1 ::
     SP(1)=0 :: SP(2)=0 :: IF S>3 THEN 1230
     ELSE DISPLAY AT(24,6)BEEP:"S A T Z E N D E " ::
     GOTO 1090
1080 GOTO 1110
1090 IF S=3 THEN DISPLAY AT(24,5):"EINEN
     ABSCHLUSSSATZ ?" ::
     ACCEPT AT(24,28)VALIDATE("JN")SIZE(-1):ENT$ ::
     IF ENT$="N" THEN 1230
1100 RETURN
1110 P(1)=1 :: P(2)=1 :: DISPLAY AT(24,1):" "
1120 DISPLAY AT(24,5):"     TIEBREAK" :: T=1 ::
     GOSUB 940 :: IF AUF=1 THEN AUF=2 ELSE AUF=1
1130 IF E$="ENDE" THEN DISPLAY AT(5,S*2+17):SP(2);::
     DISPLAY AT(6,2*S+17):SP(1); :: SP(1),SP(2)=0 ::
     S=S+1 :: P(1),P(2)=1 :: T=0 ELSE 1150
1140 DISPLAY AT(5,11):"0       ";::
     DISPLAY AT(6,11):"0       ";::
     DISPLAY AT(24,6)BEEP:"S A T Z E N D E " ::
     IF S>3 THEN 1230
1150 RETURN
1160 !ENDE DES BALLWECHSELS
1170 CALL SOUND(50,-7,0)
1180 IND=0 :: FOR I=9 TO 6 STEP -1 ::
     CALL SOUND(-1000,-5,I):: NEXT I
1190 FOR I=7 TO 17 :: CALL SOUND(-1000,-5,I)::
     NEXT I :: CALL MOTION(#1,0,0,#2,0,0,#3,0,0)
1200 GOSUB 930
1210 ON AUF GOSUB 740,830 :: GOTO 500
1220 !      SPIELENDE
1230 DISPLAY AT(24,1):"     S P I E L E N D E " ::
     FOR I=20 TO 0 STEP -1 :: CALL SOUND(-400,-6,I)::
     NEXT I
1240 FOR I=1 TO 30 :: CALL SOUND(-1000,-6,I):: NEXT I
1250 DISPLAY AT(24,5):"NOCH EIN SPIELCHEN ?" ::
     ACCEPT AT(24,26)VALIDATE("JN")SIZE(-1):R$ ::
     IF R$="N" THEN 1270
1260 DISPLAY AT(5,11):" 0{7 SPACE}0 0 0";::
     DISPLAY AT(6,11):" 0{7 SPACE}0 0 0";::
     GOTO 330
1270 END
1280 CALL HCHAR(9,10,ZUS,14)::
     CALL HCHAR(10,9,ZUS,16)::
     CALL HCHAR(11,8,ZUS,18) :: CALL HCHAR(12,7,ZUS,20)
1290 FOR J=1 TO 5 :: CALL HCHAR(13,4*J+3,112)::
     CALL HCHAR(13,4*J+4,113)::
     CALL HCHAR(13,4*J+5,112)::
     CALL HCHAR(13,4*J+6,114)
1300 IF BR=-1 THEN CALL JOYST(2,AA,BB)::
     CALL MOTION(#2,-BB*2,AA*2)ELSE
     CALL JOYST(1,AA,BB):: CALL MOTION(#1,-BB*2,AA*2)
1310 NEXT J :: RETURN
1320 !  Z E I T  - L U P E !!{20*chr(45)}!
1330 CALL SOUND(400,524,0)::
     CALL SOUND(800,1048,0,524,0,131,0)
1340 DISPLAY AT(24,3):"W I E D E R H O L U N G "
1350 FOR J=1 TO WE-1 :: CALL LOCATE(#1,SZ1(J),SS1(J),
     #2,SZ2(J),SS2(J),#3,BZ(J),BS(J))::
     CALL MOTION(#3,BRZ(J),BRS(J))
1360 CALL LOCATE(#1,INT((SZ1(J)+SZ1(J+1))/2),
     INT((SS1(J)+SS1(J+1))/2),
     #2,INT((SZ2(J)+SZ2(J+1))/2),
     INT((SS2(J)+SS2(J+1))/2))
1370 CALL POSITION(#3,YO,XO)::
     ON SGN(BRS(J))+2 GOTO 1380,1400,1390
1380 IF BS(J+1)<XO THEN 1370 ELSE 1400
1390 IF BS(J+1)>XO THEN 1370
1400 CALL SOUND(50,-7,0)
1410 NEXT J
1420 J=WE :: CALL LOCATE(#1,SZ1(J),SS1(J),#2,SZ2(J),
     SS2(J),#3,BZ(J),BS(J))::
     CALL MOTION(#3,BRZ(J),BRS(J))
1430 FOR I=1 TO 40 :: CALL SOUND(-50,I+120,10)::
     NEXT I
1440 RETURN
```

TI 99/4A ***Erweitern Sie Ihr System***

- *volle Kompatibilität zum TI 99/4A*
- *softwaremässige Anpassung*
- *alle Komponenten einzeln funktionsfähig*
- *Erweiterung durch einfaches Aneinanderstecken*

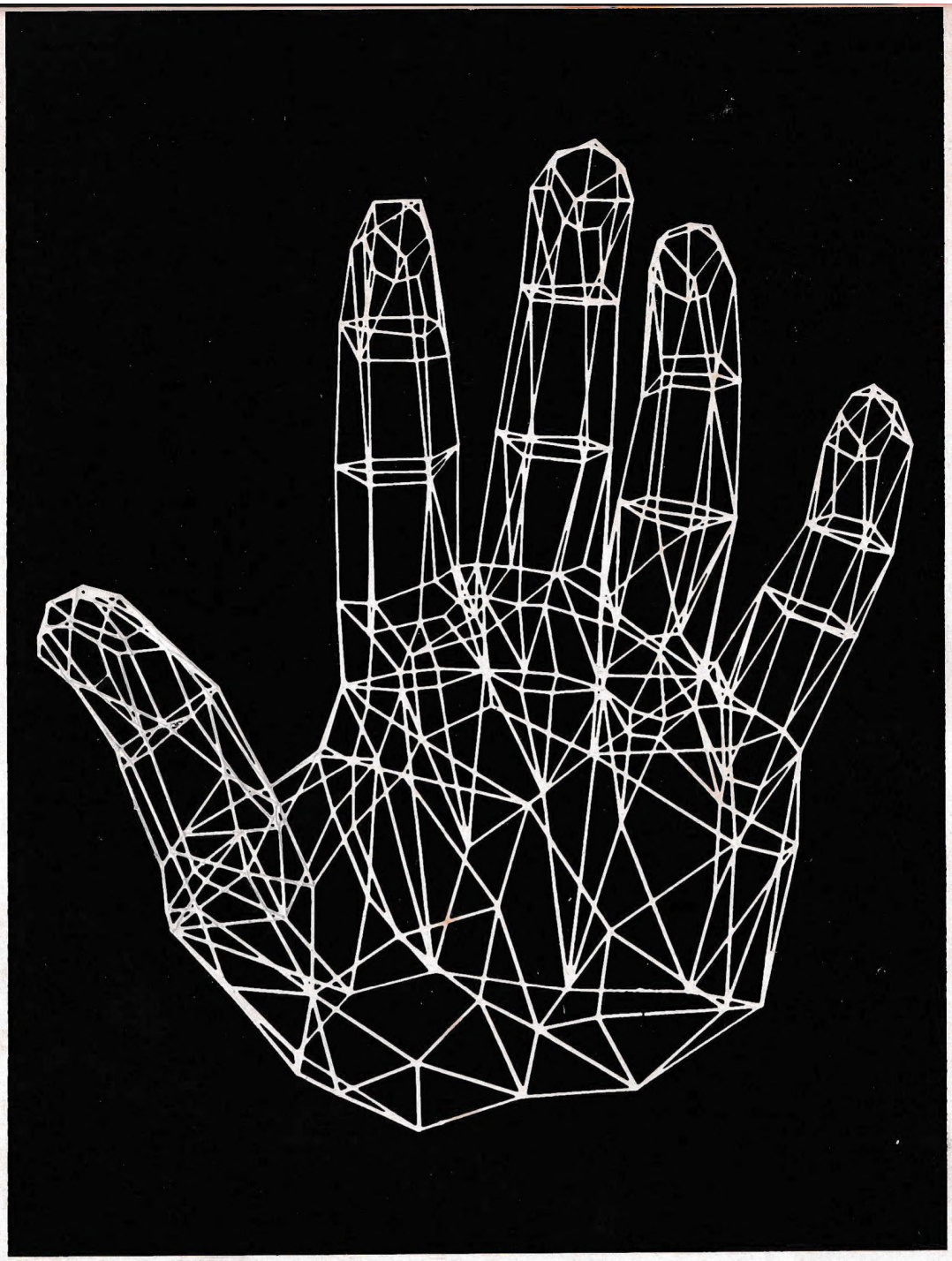
mit System!

TI 99/4A

Nutzen Sie den Telefonkontakt und fordern Sie Unterlagen an: **040 - 668 22 56**

Schnittstellen - Speichererweiterungen - Disketten-Steuerungen - Disk-Laufwerke

atronic Bunke & Vernunft oHG
Postfach 700 124
2000 Hamburg 70



CALL PLOT

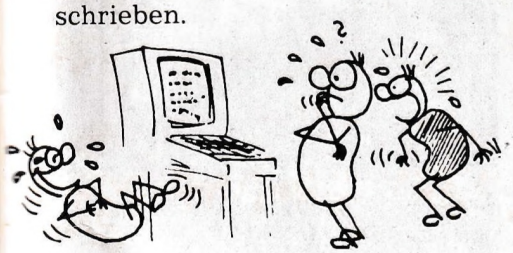
Der Texas Instruments Heim-Computer gibt sich recht verschlossen. Die spärliche Dokumentation seines Innenlebens bewirkt eine rege Nachfrage nach nützlichen Routinen und Informationen.

Wie bei so manchem TI 99/4A User hat sich das Fehlen eines Befehles, der die Ansteuerung eines einzelnen Bildschirmpunktes gestattet, auch auf meinen Gemütszustand ausgewirkt. Allerdings kam ich aus anfänglicher Mißstimmung auf ein Unterprogramm, das das Übel beseitigt und den Namen CALL PLOT trägt. Es ist ratsam, dieses Unterprogramm vor dem Schreiben des eigenen Hauptprogrammes zu laden, da der TI 99/4A mit Cassette als Massenspeicher keine Programme laden kann, ohne das im Speicher befindliche Programm zu löschen. Das Programm wird vom Hauptprogramm aus mit CALL PLOT (y,x) aufgerufen. Ein Beispielhauptprogramm ist am Ende dieses Berichtes zu finden. CALL PLOT ist mit sehr hohen Zeilen-

nummern versehen, damit es dem Hauptprogramm nicht „in die Quere" kommt.

Im Detail

In Zeile 32731 kann die Kapazität dieses Programms je nach Ihrem Bedarf gewählt werden, und zwar mit einer Zahl zwischen 1 und 110. Wählt man eine Zahl, die zu klein für seinen Bedarf ist, gibt der Computer zu dem Zeitpunkt eine Fehlermeldung aus, zu dem die Kapazität überschritten wird. Bei Wahl einer Kapazität über 52 wird das Alphabet teilweise überschrieben.

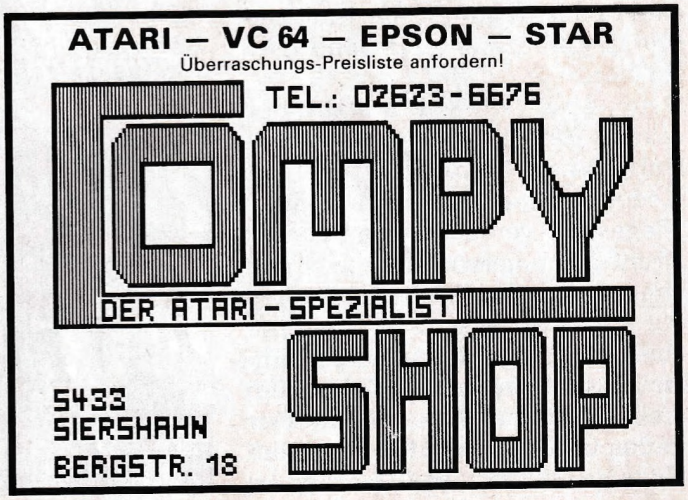


In Zeile 32740 wird abgefragt, ob sich auf der angegebenen Position bereits ein Zeichen befindet. Der in der Variablen G folgende Wert entspricht dem Wert dieses Zeichens. Ist dieses Zeichen nicht gleich dem SPACE-Zeichen (Wert von " "=32), springt das Programm zur Zeile 32747. Befindet sich jedoch auf dem bezeichneten Feld ein SPACE-Zeichen, fährt das Programm in Zeile 32742 mit der Programmausführung fort. Da jetzt ein neues Zeichen definiert wird, setzt der Rechner die Zählvariable Z um eins nach oben. In Zeile 32743 wird an die Stelle 2RY-1 der Wert des Punktes gesetzt, der geplottet werden soll. Somit ist das Aussehen des Strings bestimmt, der unseren Punkt zeichnen soll.
Als nächsten Schritt definieren wir in Zeile 32744 den neuen Character und setzen diesen Character in Zeile 32745 auf das entsprechende Feld. In Zeile 32746 folgt nun noch der Rücksprung ins Hauptprogramm. Für den Fall, daß an der bezeichneten Stelle bereits ein Character war, folgt ab Zeile 32747 ein Additionsprogramm. Das Prinzip dieses Programmes entspricht dem vorangegangenen, nur daß hier der Wert des Punktes zu dem bisher bestehenden String (Zeile 32747) addiert wird (Zeilen 32757 und 58).
Ein Hauptprogramm zu diesem Unterprogramm könnte wie folgt aussehen:

```
100 CALL CLEAR
110 FOR I=0 TO 6.4 STEP 0.1
120 X=I*36
130 Y=SIN(I)*90+95
140 CALL PLOT(Y,X)
150 NEXT I
```

Dieses Programm läuft nur auf TI 99/4 oder TI99/4A mit EXTENDED BASIC Modul. ■

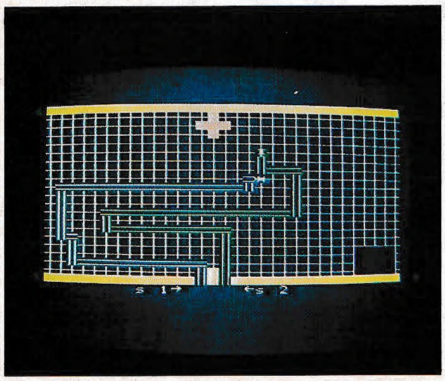
```
32730 SUB PLOT(Y,X)
32731 KAPAZITAET=100
32732 IF X<0 OR X>255 OR Y<0 OR Y>190 THEN SUBEXIT
32733 X=INT(X)
32734 Y=INT(Y)
32735 IF Z<>0 THEN 32738
32736 HEX$(0)="80" :: HEX$(1)="40" :: HEX$(2)="20" ::
      HEX$(3)="10" :: HEX$(4)="08" :: HEX$(5)="04" ::
      HEX$(6)="02" :: HEX$(7)="01"
32737 C=143-KAPAZITAET
32738 RX=X-INT(X/8)*8
32739 RY=Y-INT(Y/8)*8
32740 CALL GCHAR(INT(Y/8)+1,INT(X/8)+1,G)
32741 IF G<>32 THEN 32747
32742 Z=Z+1
32743 R$=RPT$("00",RY)&HEX$(RX)&RPT$("00",7-RY)
32744 CALL CHAR(C+Z,R$)
32745 CALL HCHAR(INT(Y/8)+1,INT(X/8)+1,C+Z)
32746 SUBEXIT
32747 CALL CHARPAT(G,B$)
32748 R$=""
32749 BE$=SEG$(B$,RY*2+1,2)
32750 IF ASC(SEG$(BE$,1,1))>64
      THEN B1=ASC(SEG$(BE$,1,1))-55
      ELSE B1=VAL(SEG$(BE$,1,1))
32751 BE2$=SEG$(BE$,2,1)
32752 IF ASC(BE2$)>64 THEN B2=ASC(BE2$)-55
      ELSE B2=VAL(BE2$)
32753 A1$=SEG$(HEX$(RX),1,1)
32754 IF ASC(A1$)>64 THEN A1=ASC(A1$)-55
      ELSE A1=VAL(A1$)
32755 A2$=SEG$(HEX$(RX),2,1)
32756 IF ASC(A2$)>64 THEN A2=ASC(A2$)-55
      ELSE A2=VAL(A2$)
32757 R1=A1+B1
32758 R2=A2+B2
32759 IF R1>15 OR R2>15 THEN SUBEXIT
32760 IF R1<10 THEN R1$=STR$(R1)ELSE R1$=CHR$(R1+55)
32761 IF R2<10 THEN R2$=STR$(R2)ELSE R2$=CHR$(R2+55)
32762 Y$=R1$&R2$
32763 IF Y$=BE$ THEN SUBEXIT
32764 R$=SEG$(B$,1,2*RY)&Y$&SEG$(B$,2*RY+3,14-RY*2)
32765 CALL CHAR(G,R$)
32766 CALL HCHAR(INT(Y/8)+1,INT(X/8)+1,G)
32767 SUBEND
```

SOFTWARE IM TEST

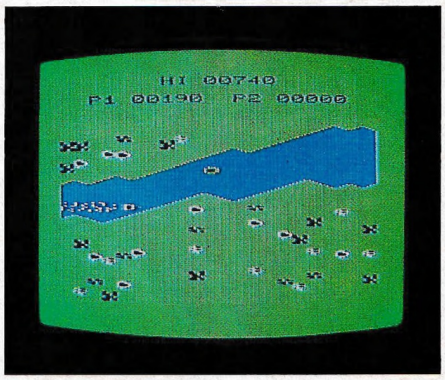
TRON, ANGLER und SAFARI

Das Spiel TRON wurde vom legendären Computer-Motorrad-Rennen des gleichnamigen Filmes abgeleitet. Es ist ein Spiel für zwei Personen, bei dem jeder ein Motorrad zu steuern hat. Beim Fahren hinterlassen beide Fahrzeuge eine Energiewand auf dem Kampfraster, mit der das andere Motorrad nicht in Berührung geraten darf. Ein Unfall wäre die Folge, der das Aus bedeuten würde. Auch eine Kollision mit der Außenwand oder einem Hindernis führt zum Spielende. Das Spiel ist für eine Terminal-Bedienung geschrieben, und man müßte erst selbst das Programm ändern, um mit Joysticks spielen zu können. Die Schnelligkeit und Beweglichkeit der Motorräder lassen einiges zu wünschen übrig, und leider ver-



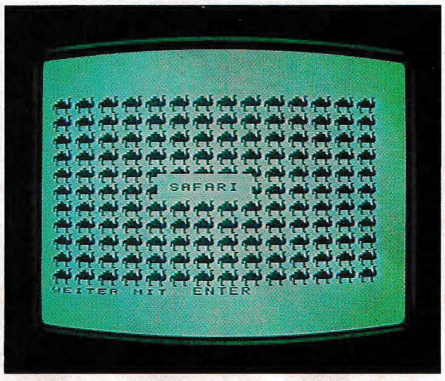
führt auch die simple und langweilige Graphik nicht zum häufigen Spielen.

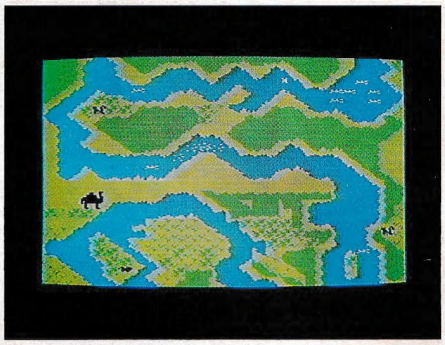
Beim Spiel ANGLER sieht der einzige Spieler von der Seitenansicht seinen Angler, die Angel mit Köder und die Fische, die je nach „Tageszeit" verschieden aussehen. Er muß nun versuchen, so schnell wie möglich die Fische zu fangen, auf die es leider auch gefährliche Haie abgesehen haben. Der Angler hat seinerseits die Möglichkeit, die Angelrute am Ufer oder weiter in den See auszuwerfen. Auch die Höhe des Köders im Wasser kann

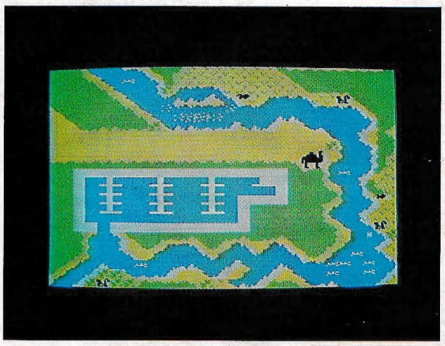


verändert werden. Im Gegensatz zu TRON muß hier der Joystick verwendet werden. Es ist bei diesem Spiel nicht möglich, die Angel schnell zu bewegen; auch schwimmen die Fische nach jeder Stufe nur einmal davon. Außerdem fehlen jegliche Schwierigkeitsabstufungen. Lediglich die Farbgraphik ist noch der Rede wert. Sie ist deshalb interessant, weil eine Nacht- und Tagdarstellung vorhanden ist. Nachdem 85 Fische gefangen sind, ist das Spiel beendet. Das allerdings zu schaffen verlangt einem viel Zähigkeit ab und ein wiederholtes Spielen noch mehr davon.

SAFARI kann ebenfalls nur von einer Person gespielt werden. Man befindet sich auf einer Fotosafari in Afrika und fährt einen sich schlängelnden Fluß hinunter, um nach Kwabungi-Town zu gelangen. Es muß ein Floß gesteuert werden, Krokodilen ist auszuweichen. Wenn man an einem interessanten Tier vorbeikommt, muß der Auslöser der Kamera gedrückt werden und man erhält Punkte.







Anderenfalls schrumpft bei jeder verpaßten Gelegenheit das Punktekonto. SAFARI wird wieder, wie TRON, auf der Tastatur gespielt. Das Floß ist sehr unbeweglich und die Graphik kann nicht überzeugen.

Der Kauf der Kassette lohnt nur, wenn man als Besitzer des TI 99/4a nicht so viel Geld investieren möchte, wie für Modulversionen nötig wäre. Sollte jemand noch keine Joysticks besitzen, kann er lediglich zwei Spiele (TRON und SAFARI) spielen.

Hersteller: Radix
Bezugsquelle: Radix

Graphik:

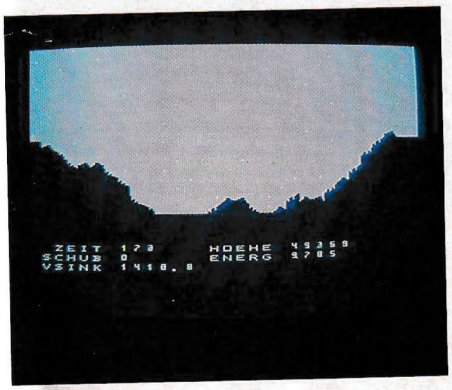
Sound:

Spielidee:

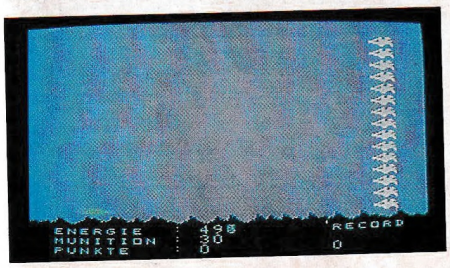
Testergebnis:

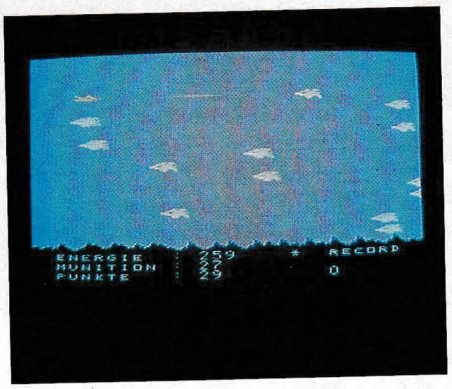
MONDLANDUNG + SPACE SPIONAGE

Die Spiele auf dieser Kassette sind für TI und EXTENDED BASIC geschrieben. Mondlandung ist das bekannte Spiel, bei dem der Bordrechner versagt und man auf die Handsteuerung des Raumschiffes angewiesen ist. Eine Anzeige führt dem Spieler am Anfang des Spiels seine immense Fallgeschwindigkeit vor Augen. Nun hat man die Möglichkeit, das Raumschiff durch Schubkorrekturen mit den Zifferntasten des Computers zu steuern.
Das Spiel hat eine ansprechende Graphik, doch es fehlt die Spannung, da man nur auf seine Untertasse und den Mond zu achten hat und schon bald keine unerwarteten Dinge mehr geschehen.



In SPACE SPIONAGE soll man sich als Commander eines Sicherungsraumschiffes fühlen. Die Aufgabe besteht darin, einen bestimmten Geländeabschnitt des Planeten Mars vor den Blicken neugieriger Raumspione zu schützen. Die feindlichen Flugkörper nehmen nicht die Jagd auf den Verteidiger auf, sondern





sie werden so zahlreich, daß man gezwungen ist, sie abzuschiessen. Dabei hat man aber nur eine begrenzte Anzahl von Munition und muß mit ihr sparsam umgehen.
In diesem Spiel gibt es Schwierigkeitsgrade, die auch einen guten Spieler durchaus herausfordern können. Die geringe Flexibilität des Flugzeugs läßt leider keine Akrobatik auf dem Fernseher zu. Alles in allem dennoch zwei unterhaltsame Spiele. Beim Kauf ist allerdings zu bedenken, daß SPACE SPIONAGE Joysticks und beide Spiele das Modul Extended Basic benötigen.

Hersteller: Radix
Bezugsquelle: Radix

Graphik: ☹
Sound: ☹
Spielidee: 😐

Testergebnis:

☹

Vom Hersteller

Dieses Buch, herausgegeben vom TEXAS INSTRUMENTS Learning Center, ist eine regelrechte systematische Bedienungsanleitung für den TI 99/4A. Hier wird vom Einschalten des Rechners über die Direkteingabe bis zum Programmieren in Extended-BASIC alles ausführlich erklärt. Man hat es mit einem kompakten Paket von Arbeitsunterlagen zu tun, das dank einer sinnvollen Gliederung leicht überschaubar ist. Grafiken und Schirmbild-Kopien wurden gut eingesetzt.
Zum ausgezeichneten Nachschlagewerk wird es durch ein gut erklärtes Verzeichnis aller BASIC-Befehle, ausführlich erläuterte Fehlermeldungen und ein vorbildlich zu nennendes Register.
Die Druckvorlagen für dieses Buch wurden auf einem TI-Printer erstellt (vielleicht, um dessen Leistungsfähigkeit unter Beweis zu stellen), jedoch hätte eine Satzschrift das Lesen etwas angenehmer gemacht. Doch diese leise Kritik kann unseren Eindruck nicht trüben, daß hier für jeden TI-Benutzer eine echte Programmierhilfe geboten wird. ■

Hrsg.: TI Learning Center
48.- DM
ISBN 3-88078-039-0

Commodore meldet ein Umsatzplus von 103 % gegenüber dem Vorjahr. Seine Gewinne stiegen gar um 116 % bezogen auf die Monate Juli bis September 1983 und 1982. Können uns optimistische Zahlen wie diese beruhigen?

Ist nach Texas Instruments Rückzug vom Heim-Computer-Markt und Hektik des Weihnachtsgeschäfts die Computer-Welt wieder in Ordnung? Oder war das Aus für den TI 99/4A nur Auftakt für weitaus sensationellere Ereignisse, die wir in diesem Frühjahr noch erleben werden?
Herbe Meldungen brachten im letzten Jahr Unruhe unter uns Heim-Computer-User. ZX 81-Fans grübeln nicht nur über neuen Listings, seit bekannt wurde, daß Sinclair in den USA von ernsthaften Absatzsorgen geplagt wird (ece 1, S.62). Atari-Freunde sind nachdenklich gestimmt, seit der Stern am 20. Oktober herausposaunte, daß Atari dieses Jahr Verluste nahe einer Milliarde wegstecken mußte und 3000 seiner 10 000 Mitarbeiter vor die Tür setzte. Und den TI 99/4A-Besitzern mag glatt der Joystick aus der Hand gefallen sein, als sie erfuhren, daß sich Texas Instruments vom Heim-Computer-Markt verabschiedet. Klar, daß Commodore-Trickser ein nervöses Kribbeln in der Magengrube bekommen, wenn sie für ihr Gefühl einmal ungewöhnlich lange auf den längst bestellten Monitor warten müssen. Die Reihe der Nachrichten, die keineswegs zur Erheiterung unserer Gemüter beitragen, läßt sich zu unser aller Überdruß auch noch fortsetzen. Bei 'Osborne' (Personal-Computer) hat der Konkursverwalter die Aufsicht übernommen, 'Victor' (Sirius) (gleichfalls Personal-Computer) hat die Hälfte seiner Mitarbeiter entlassen, und von „Microflop-

py-Krieg" ist die Rede. (Markt & Technik Nr. 38, 23.9.83; Nr.39, 30.9.; Nr.37, 16.9.) Die Folge: Große Verunsicherung! Was ist los? Sollte etwa die ausgefeilte Technik, die wir für unser gutes Geld erworben haben, ihre Zukunft verlieren?

Der Fall TI Ein Lehrstück für alle User

Betrachten wir die Vorfälle bei Texas Instruments einmal genauer. Der TI 99/4A ist durchaus keine alte Gurke, für die es Zeit wurde, vom Markt zu verschwinden. Nicht umsonst fiel TI 1983 ein Fünftel des US-Heim-Computer-Marktes zu, der allein die Hälfte des Weltmarkts ausmacht. Eigentlich also ein rundherum erfolgreiches Produkt, dieses Grundgerät. Es hatte anscheinend nur einen einzigen Fehler: Es war zu billig geworden! In den USA fiel sein Preis in den 2½ Jahren bis zum April 1983 um 71 % von 525 Dollar auf 149 Dollar. Zwischen den Computer-Anbietern war der Preiskrieg ausgebrochen! Jeder versuchte jeden zu unterbieten, um sein Produkt besser zu verkaufen und den anderen am Markt ausstechen zu können. TI war nun der erste Anbieter, der sich von diesem Kampfgeschehen zurückzog. Am 4. November 1983 erklärte TI:

„Aufgrund der weltweiten ungenügenden Geschäftsentwicklung im Heim-Computer-Bereich, die im III.Quartal 1983 zu erheblichen Verlusten im Consumer-Bereich geführt hat, sowie der unsicheren weiteren Entwicklung im Heim-Computer-Markt haben wir beschlossen, uns aus diesem Markt zurückzuziehen." Eine Nachricht wie ein Blitz aus scheinbar heiterem Himmel. Kaum war der erste Schreck halbwegs verdaut, da teilte TI auch schon 4 Tage später mit, sein Lager an Peripherie-Geräten sei leer. Sollen wir dies für Zufall halten? Wäre eine frühzeitigere Information der Kunden nicht möglich gewesen? Und schon geistern umgebaute englische und amerikanische Konsolen über die Märkte, die im Bestimmungsland keine Abnehmer mehr finden und für die TI jeglichen Service ablehnt. Überhaupt tat sich TI mit der Informationspolitik gegenüber seinen Kunden äußerst schwer. Die Fernsehspots für den 99/4A liefen nach dem 4.11.1983 munter weiter - als „Service für die Kunden" (so TI) - statt die teuer angemietete Zeit zur Aufklärung der verunsicherten 99/4A-Besitzer zu nutzen. Ebenso wenig ist es gentlemanlike, über den 4. November hinaus Werbeprospekte über ein Angebot zu verschicken, das überhaupt nicht mehr besteht, „Peripherie die Sie schon lange haben wollten", die wir aber tatsächlich von TI längst nicht mehr bekommen. Soll man sich da etwa nicht geprellt fühlen?

Die New Yorker Börse belohnte TI für den „mutigen" Schritt fort vom gefährlichen Heim-Computer-Markt mit einem erheblichen Anstieg des Aktien-Kurses. Doch damit ist den 99/4A-Besitzern wenig gedient. COMPUTER-Praxis wird versuchen, euch eine praktische Hilfe zu sein. Wir werden die Augen offen halten, welche Fremdanbieter die durch TI's Rückzug im Peripheriebereich klaffende Lücke schließen werden. Durch fortgesetzte Berichterstattung in unsererm TI-Teil werden wir uns um eure Interessen bemühen. Einen ersten Schritt in diese Richtung haben wir mit unserem Artikel zum Hersteller 'atronic' in diesem Heft unternommen.

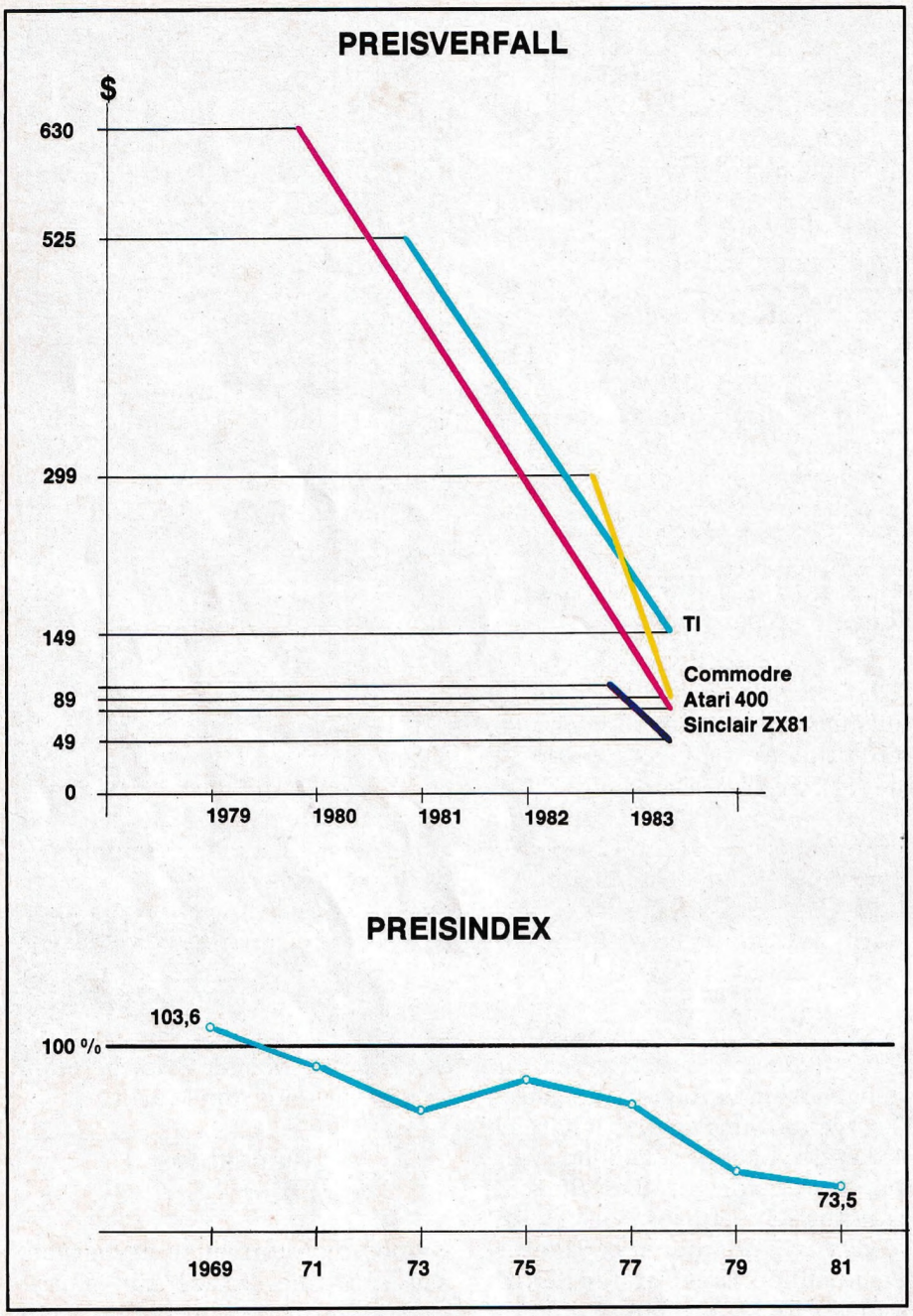
Wichtig für uns alle ist es jedoch zu verstehen, warum sich TI zu einem so unpopulären Schritt entschloß.

Hätte TI weitermachen können?

Entscheidungen in bedeutenden Unternehmen wie TI werden kaum ohne gründliche Überlegungen getroffen. Wir können also annehmen, daß für TI weit mehr auf dem Spiel stand, als seine Heim-Computer-Kunden zu verlieren. Es ging um den guten Ruf des angesehenen Unternehmens. Nachdem die Preise für Heim-Computer immer weiter in den Keller fielen, war für TI mit dem 99/4A nichts mehr zu verdienen. Er war mehr wert, als er kostete. Das ist der Grund für TI's Verluste in Millionenhöhe. Vielleicht wären sie noch eine Weile aufzufangen gewesen - durch Gewinne, die zum Beispiel der Taschenrechnermarkt einbringt. Doch scheint es dem Unternehmen nicht vertretbar gewesen zu sein, die Taschenrechnerkunden für die Heim-Computer mitbezahlen zu lassen. Wäre also für TI noch die Möglichkeit geblieben, die Herstellungskosten für den 99/4A zu senken. Es scheint jedoch mit dem Preissturz inzwischen eine Grenze errreicht, von der an Einsparungen auf das produzierte Gerät durchschlagen. Spart man zum Beispiel an der Schlußkontrolle vor Verlassen des Werkes, so wirkt sich das direkt auf die Qualität der Computer aus. TI's unternehmerisches Ziel ist es aber seit je, qualitativ hochwertige Geräte anzubieten. Dafür steht der Name. Einziger Ausweg blieb so der Rückzug vom Markt.

Was ist los mit dem Markt?

Keinem von uns blieb verborgen, wie sehr die Preise für Heim-Computer in den Keller gesackt sind. Eine Preislawine geht zu Tal, daß es einem die Sprache verschlägt. Sicher: Man kennt langfristige Preisentwicklungen nach unten auf Gebieten, auf denen technischer Fortschritt vorangetrieben wird. Neue Erfindungen und technische Verbesserungen schlagen auf das Preis/Leistungs-Verhältnis durch. So etwas läßt sich durch die Jahre hindurch messen. Eine solche Preisentwicklung zeichnet sich zum Beispiel ab, wenn man sich die Preise für Erzeugnisse der Computer-Industrie anschaut:



Wir haben es hier mit einer recht ausgeglichenen Abwärtsbewegung zu tun. Man könnte Sprünge aufgrund von bedeutenden technischen Fortschritten erwarten (die ersten Minicomputer erschienen 1970 auf dem Markt, Micros verbreiten sich seit 1980 unaufhaltsam). Aber sie treten auch deswegen nicht auf, weil Preise den verschiedensten Einflüssen unterliegen. Produktentwicklung kostet zunächst mal Geld, ebenso der Aufbau einer Vertriebsstruktur. Die Serviceleistungen muß der Käufer beim Geräteerwerb mitbezahlen, ebenso wie die Kosten für die Werbung. Was sich jedoch in der letzten Zeit auf dem Markt für Heim-Computer abspielt, läßt sich mit einem normalen Kurvenverlauf kaum noch zur Deckung bringen. Worüber wir Käufer uns im ersten Moment freuen, daß nämlich im Vorreiterland in Computerdingen die Preise in einem Maße purzelten, daß man auch noch in der BRD auf weitere Preissenkungen „hoffen" kann, verschlägt uns im zweiten Moment das Grinsen. Was wir beobachten, ist nicht der Erfolg voranschreitender Technik, sondern Ergebnis eines gnadenlosen Preiskriegs und Verdrängungswettbewerbs.

Was bedeutet Preiskrieg für uns Kunden?

Die Anbieter drücken ihre Preise mit aller Macht herunter, um ihre Mitbewerber in die Knie zu zwingen. Wer da nicht mitziehen kann, dessen Geräte drohen unverkauft zu bleiben.

Uns Kunden droht aber weitaus Schlimmeres. Unternehmensstrategien, die nur noch auf problemlosen und massenhaften Absatz ausgerichtet sind, gefährden unsere Interessen. Uns nützen billigste Preise überhaupt nichts, wenn sie nur noch den nackten Computer betreffen. Wir fordern vom Hersteller Service, den niemand außer ihm leisten kann.

Wir brauchen Geräte, die auf Herz und Nieren geprüft aus der Fertigung kommen. Nicht wir wollen erst zu Hause überprüfen, ob unser Computer tatsächlich auch in einwandfreiem Zustand ist, nachdem wir ihn bezahlt haben. Wie niedrig der Preis auch sein mag, für uns ist Geld immer noch Geld.

Was wir ebenso nötig brauchen, sind ausführliche und verständliche Bedienungsanleitungen, aus denen nicht nur zu entnehmen ist, wie man den Netzstecker in die Steckdose einführt. Und wenn einmal Reparaturen nötig sind, so sind wir darauf angewiesen, daß sie die Hersteller in vernünftiger Zeit durchführen, auch wenn das für sie ein aufwendiges Geschäft bedeutet. Wir wollen, daß unser System weiterentwickelt wird, daß neue Peripherie gebracht wird, die uns weitere Anwendungsmöglichkeiten erschließt. Und wir brauchen die Gewähr, daß Zusatzgeräte und Ersatzteile auch immer schnell erhältlich sind.

Vielleicht können die Hersteller ihre Preise noch weiter senken, wenn sie in all diesen Punkten kräftig sparen. Eine Fortsetzung

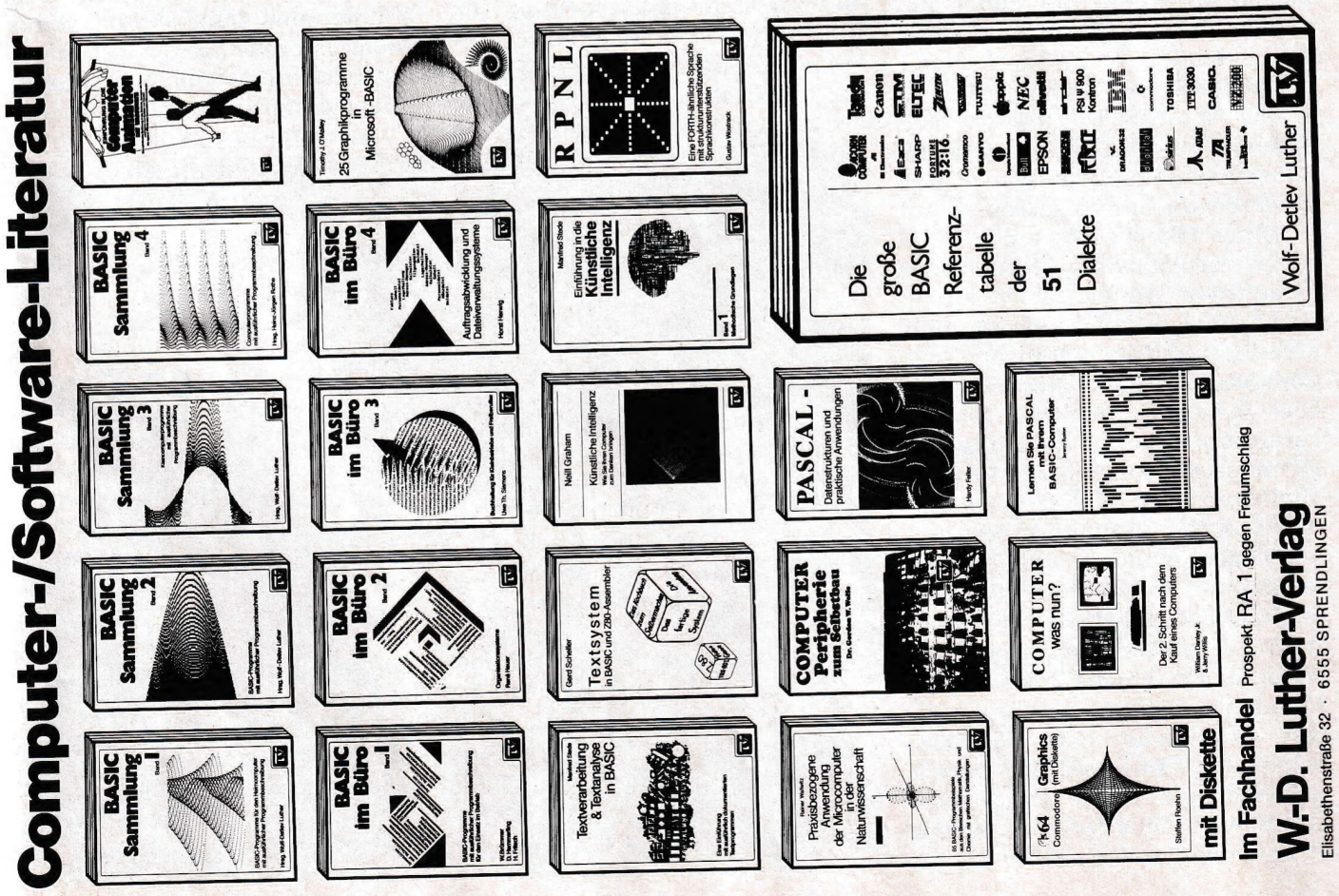
dieses Preiskriegs dürfte jedoch unvermeidlich schlimme Folgen für uns haben.

Was aber machen die Hersteller?

Blufft Alwin Stumpf, Leiter Vertrieb und Marketing bei der Commodore-Büromaschinen GmbH, wenn er verkündet, daß jedes neue Computer-Produkt, egal zu welchem Preis es auf den Markt kommt, in jedem Fall so kalkuliert ist, daß es notfalls in einem Jahr auch für 69 Dollar angeboten werden kann (Markt & Technik Nr. 42, 21.10.83, S. 74)? Wäre ein solcher Preisverfall überhaupt zu verantworten? Natürlich setzen solch niedrige Preise riesige Umsatzmengen voraus, damit der geringe Gewinn, den die Hersteller pro verkauftem Stück noch ziehen, überhaupt eine angemessene Summe erzielt. Ein gutes Beispiel dafür ist 'Apple', dessen Umsätze zwar von 175,8 auf 273 Mio. Dollar stiegen, gleichzeitig aber die Gewinne von 18,7 auf 5,1 Mio. Dollar fielen (Markt & Technik Nr. 45, 11.11.83). Mit dem Gewinn pro Stück für den Hersteller sinkt aber auch gleichzeitig die Gewinnspanne für die Händler. Atari kalkuliert schon jetzt, daß eigentlich nur Geräte mit einem Preis über 600 DM für Warenhäuser und Computerläden interessant sind. Bei Computern der unteren Preisklassen muß ein Händler enorme Stückzahlen absetzen können, um nennenswerte Gewinne zu erwirtschaften. Das können aber momentan nur Kettenläden und Versandhäuser bieten. Müssen wir uns also schon mal darauf einstellen, demnächst im Kaffee-Depot Rat zu suchen, wenn wir uns einen Plotter zulegen wollen? Im Personal-Computer-Bereich wird sogar allen Ernstes mit dem Gedanken gespielt, bei einträglichem Verkauf von Software-Paketen die Hardware als Gratisdreingabe mitzuliefern (Markt & Technik, 42, 21.10.83).

Kein Ende des Preiskrieges in Sicht?

Den Computer - Herstellern scheint es um einiges zu gehen, wenn sie bereit sind, mit derart harten Bandagen zu kämpfen. Heim-Computer sind nicht deswegen etwas so umwälzend Neues, weil man auf einmal mit ihnen nun in der Lage ist, Schiffeversenken nicht nur mit Papier und Bleistift unter der Schulbank während langweiligen Unterrichts zu spielen, sondern jetzt auch zu Hause allein bei den Schulaufgaben, oder weil man sich jetzt Knabberkram und Limo für den Schachpartner sparen kann. Die



Sache ist vielmehr die, daß unsere Heim-Computer zur vierten Generation der elektronischen Rechenmaschinen gehören. Während man in den 50er Jahren mit Röhren, Anfang der 60er dann mit Transistoren herumpuzzelte, setzte mit der Erfindung der integrierten Schaltungen (IC's) die dritte Generation ein. Von Stufe zu Stufe wurden der Platz, den man zum Aufstellen der Apparaturen benötigt, und die Kosten der Herstellung geringer. Entscheidend war der Durchbruch zur vierten Generation, die mit Siliziumelementen (L.S.I.) bestückt ist, und der 1975 erfolgte. Die Erfindung des IC's und der Chips gehen übrigens auf das Konto der Texas Instruments-Ingenieure. Nun war eine Verkleinerung der Speicher und Rechnerelemente erreicht, die dem Computermarkt völlig neue Blickwinkel eröffnete. Die Miniaturisierung war soweit vorangeschritten, daß die neue Computergeneration in jeder Aktentasche, erst recht in jedem Haushalt bequem unterzubringen war. Auch der technlogische Stand der Fertigung von Chips erforderte geradezu die Produktion großer Stückzahlen, wenn sie einen hohen Grad von Wirtschaftlichkeit erzielen sollte. Und noch etwas begann um diese Zeit aufzuscheinen: Die Utopie einer Computerisierung der Gesellschaft schien zum Greifen nah.

Nicht nur die Warenproduktion könnte durch den Einsatz von Computern rationalisiert werden. Die Verwirklichung der Datenverarbeitung in jedem trauten Heim setzt weitaus mehr Möglichkeiten frei. Durch Verkabelung von mit Computern bestückten Haushaltungen könnte die technische Revolution, die momentan auf der Ebene von Betrieben abläuft, enorm ausgedehnt werden. Es bedarf nur wenig Phantasie, sich vorzustellen, wieviel Arbeitsplätze in Heimarbeitsplätze umwandelbar wären. Sachbearbeiter und Sekretärinnen, soweit nicht durch geeignete Programme ersetzbar, erledigen ihre Arbeit zu Haus am Bildschirm. Büromieten können eingespart werden, für alle Tätigkeiten am Computer kann ein Lohn bezahlt werden, den die Maschine nebenher genau auf die Leistung bezogen auf Heller und Pfennig ausrechnet. Pinkelpausen und Büroschlaf brauchen nicht mitbezahlt zu werden. Andererseits könnte man jeden computerisierten Job auf der Welt annehmen, ohne sich aus den eigenen vier Wänden zu bewegen. Auch die Zahl der Verkehrsunfälle könnte vermindert werden, wenn sich nicht mehr Millionen von Autofahrern zur selben Stunde auf den Weg zur Arbeit machten.

In wie weit diese Phantasien jemals Wirklichkeit werden, spielt hier keine Rolle. Festzuhalten bleibt, daß die Computerindustrie allen Grund hat, einen wahrhaft riesigen Absatzmarkt in der Zukunft zu wittern.

Wie können wir unsere Interessen wahren?

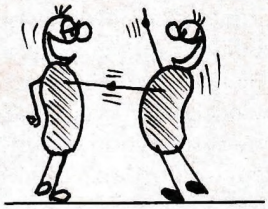
Um mit einem völlig aus den Fugen geratenen Markt zurechtzukommen, ist die einzige Möglichkeit für uns User, kühlen Kopf zu bewahren. Bevor sich jemand zum Einstieg in die Computer-Szene entschließt, sollte er sich zunächst klar darüber werden, was für Pläne er in absehbarer Zeit mit seinem neuen Gerät hat. Wenn er weiß, ob er einen Spielpartner sucht oder weitergehende Aufgaben für den Computer vorsieht, sollte er sich eingehend informieren, welche Auswahl von Geräten seinen Bedürfnissen am nächsten kommt. Dann sollte er sich den Satz aus Grundgerät und Peripherie zusammenstellen, den er für seine Zwecke benötigt und schließlich größten Wert darauf legen, daß er den gesamten Geräteset beim Kauf im funktionsfähigen Zustand nach Hause tragen kann. Der Preis sollte hier eine zweitrangige Rolle spielen und nur in Verbindung mit den Service-Leistungen gesehen werden. Billig sind nahezu alle Anbieter, hält man sich vor Augen, welche technische Leistung geboten wird. Teurer kommt es auf alle Fälle, irgendwann feststellen zu müssen, daß man dieses oder jene heißbegehrte Gerät überhaupt nicht mehr bekommt.

Für die Besitzer eines Grundsteins im Computer-Ausbausystem gilt Ähnliches: Sich frühzeitig Gedanken machen und sich früh informieren, in welche Richtung man ausbauen will und kann. Natürlich bleiben für den User, der diese Überlegungen beherzigt, immer noch Probleme. Auf sich allein gestellt kann er kaum überblicken, welche Schwierigkeiten mit Reparaturen auf ihn zukommen mögen. Als Anfänger wird er sich kaum ein Urteil über Bedienungsanleitungen bilden können. Von Lieferengpässen wird er oft nur Gerüchte zu hören bekommen. COMPUTER Praxis will darum fortlaufend das Feld des Service speziell für den deutschen Markt unter die Lupe nehmen und dem User bei seinen wichtigsten Entscheidungen mit Testberichten und fundierten Nachrichten zur Seite stehen.

Wie sieht die Zukunft aus?

In die Zukunft blicken können wir sicher nicht. Zwar hat sich die Marktlage momentan etwas entspannt. Ataris Erhöhung der Händlerpreise um 17 bis 29 % zeigt das an. Ob Commodore wirklich gefestigt aus dem Preiskrieg hervorgegangen ist, müssen neue Bilanzen zeigen. Ebenso muß sich erweisen, ob Commodore es wirklich schafft, die angewachsene Gerätenachfrage zu befriedigen. Das eingetretene Gleichgewicht nach TIs Rückzug ist jedenfalls noch recht wackelig. Die Ursachen für den Kampf um Marktanteile bestehen weiterhin. Die Dinge können jederzeit wieder in Bewegung geraten, sobald sich ein Anbieter auch nur stark genug fühlt, den Preiskrieg erneut zu erklären.■

Mit TI-Peripherie nach oben



Aus den USA kamen E.T., Aerobic und Rap. Was genauso herüberkam, war die Idee und vielleicht auch der Mut zum Computerbau in der Garage. Apples Geschichte ist schon eine Legende und auch in Deutschland haben die meisten dieser Kleinfirmen zumindest den Sprung aus der Garage geschafft.

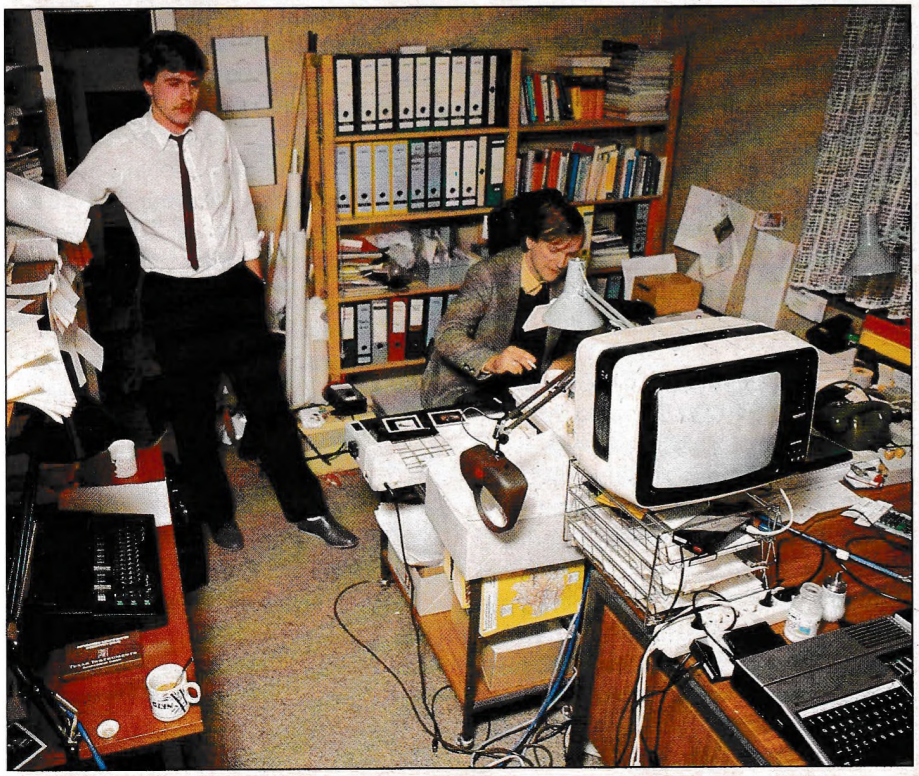
Was das Leben des Heim-Computerbesitzers in den Staaten so angenehm macht, ist das Vorhandensein enorm vieler kleiner Anbieter für Peripherie und Zubehör, die dem Anwender oft die exotischsten Nutzungsmöglichkeiten bieten. Wir wollen in Zukunft an dieser Stelle Firmen vorstellen, die mit Tatkraft und Phantasie dazu beitragen, den deutschen Heim-Computermarkt aus der Abhängigkeit von den „Großen" zu befreien.

Profitechnik für Heimcomputer

In dieser Ausgabe stellen wir die Firma „atronic" in Hamburg vor. Die ersten Erfahrungen sammelten die beiden jungen Inhaber bei der Herstellung von industriellen Steuerungen für Arbeitstische etc.; die Produktion in diesem Bereich soll auch in Zukunft wieder aufgenommen werden. Heute jedoch sind die Herren Vernunft und Bunke dabei, ihrer Firma durch die Entwicklung von Peripheriegeräten für Heim-Computer einen Namen zu schaffen. Hauptsächlich sind das die unten näher beschriebenen für den Texas Instruments TI 99/4A. Jetzt darf man sich allerdings bei der Bezeichnung „Firma" kein riesiges Gebäude mit Werkshallen und einem modernen Verwaltungstrakt vorstellen. Das Gegenteil beweisen die Fotos des Büros auf dieser Seite. Einen Einblick in die verschiedenen Produktionsstätten, die über den Norden Hamburgs verstreut liegen, konnte man der Redaktion von COMPUTER Praxis jedoch nicht geben. Es war Freitag Nachmittag, als wir unser Gespräch führten und eine Fahrt dorthin lohne sich nicht, da niemand mehr da sei.

„Sicherlich haben wir bei der Gründung von atronic ein bißchen über den großen Teich geschielt, wie die Amerikaner so etwas aufziehen", erklärte uns Olaf Vernunft. Erfolgreiche Unternehmen wie 'Apple' und Sinclair dürften dabei als Vorbilder gedient haben. Daß Herr 'Apple' unterdessen sein Geld als Produzent von Rockgruppen verdient oder möglicherweise ausgibt, trübt dabei keineswegs den Optimismus der beiden Jungunternehmer.

Die Herren Vernunft und Bunke werden die Erfolge der obengenannten Männer wohl kaum wiederholen können, dennoch dürften sie mit der bisherigen Entwicklung von atronic sehr zufrieden sein. Uns wurde versichert, daß nach nur einjähriger



Die Inhaber der Firma „atronic", Herr Vernunft (links) und Herr Jansen, erläutern uns die Produkt-Palette.

Produktion bereits schwarze Zahlen geschrieben werden. Geplant ist deshalb, in absehbarer Zukunft weitere Investitionen zu tätigen, was natürlich zur Schaffung von neuen Arbeitsplätzen führt. Gesucht werden vor allem hochqualifizierte Kräfte aus der Elektronik-Branche mit entsprechender Berufserfahrung. Es wird sogar daran gedacht, Ausbildungsplätze im technischen, sowie im kaufmännischen Bereich anzubieten. Des weiteren will die Firma einigen Studenten die Möglichkeit geben, ein Praktikum zu absolvieren.

Der TI kann sich wieder ausbreiten

Neben Speichererweiterungen für den Sinclair ZX 81 (die, wie auf dem Bild zu sehen, auch als Bausatz erhältlich sind) stellt atronic heute, außer Kleinserien für den industriellen Bereich, hauptsächlich Peripherie-Geräte für den TI 99/4A her. Diplom-Ingenieur Wolfgang Jansen ist der Verantwortliche für die technischen Entwicklungen. Mit ihm gingen wir die Palette der von atronic angebotenen TI-Peripherien durch.

1. Drei Speichererweiterungen:
   - ▷ Das einfache 32 KB-RAM Modul
   - ▷ Das 32 Kbyte CMOS-RAM, sehr viel niedriger im Stromverbrauch
   - ▷ Das 32 Kbyte CMOS-RAM Batterie-gestützt, in der Programme auch ohne Netz bis zu einem Jahr gespeichert werden können.

Eine vierte Speichererweiterung mit 64 Kbyte Kapazität und einer integrierten parallelen Schnittstelle kommt diese Tage auf den Markt.

2. Die Schnittstelle RS232C :
   - ▷ Einen Test findet Ihr im TI-Teil dieser Ausgabe.
3. Ein Disketten-Controller:
   - ▷ Dieses Steuergerät ist eine völlige Neuentwicklung, die Anfang März auf den Markt kommen soll. Der Beschreibung zufolge dürfte es ein sehr flexibles Gerät sein.
   - ▷ Es kann sowohl das alte TI-Einseiten-Laufwerk mit 90 KB Kapazität und einfacher Datendichte benutzt werden als auch ein modernes Zweiseiten-Laufwerk mit 500 KB bis 1MB Kapazität und doppelter Datendichte.
   - ▷ Das Disk-Manager-Programm wird voraussichtlich eingebaut.
4. Ein EPROM-Brenner (in der Entwicklung):
   - ▷ Das Gerät ermöglicht u.a. die Herstellung von eigenen Modulprogrammen.
5. Extended BASIC-Modul:
   - ▷ Ist ein Nachbau des TI-Extended BASIC-Moduls und
   - ▷ ist nötig, um die Peripherie steuern zu können (z.B. 32 KB-CMOS-RAM).
6. Ein Telefon-Modem:
   - ▷ Gerät zur Übertragung von Daten und Programmen über das Telefonnetz. Ausführung zunächst als Akustikkoppler, d.h. der Telefonhörer wird auf das Gerät gelegt.
7. Ein Bus-Verlängerungskabel:
   - ▷ Das ausgebaute System nimmt sehr viel Platz ein. Mit Hilfe der Busverlängerung ist das Aufstellen der Peripheriegeräte an einem Platz möglich, wo sie weniger stören als direkt neben der Konsole.

Alle in diesem Abschnitt erwähnten Geräte werden im Systemteil dieser und der folgenden Ausgaben von COMPUTER Praxis noch ausführlich getestet.

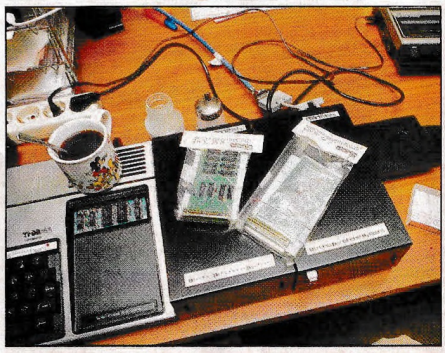
Auf Erfolg programmiert

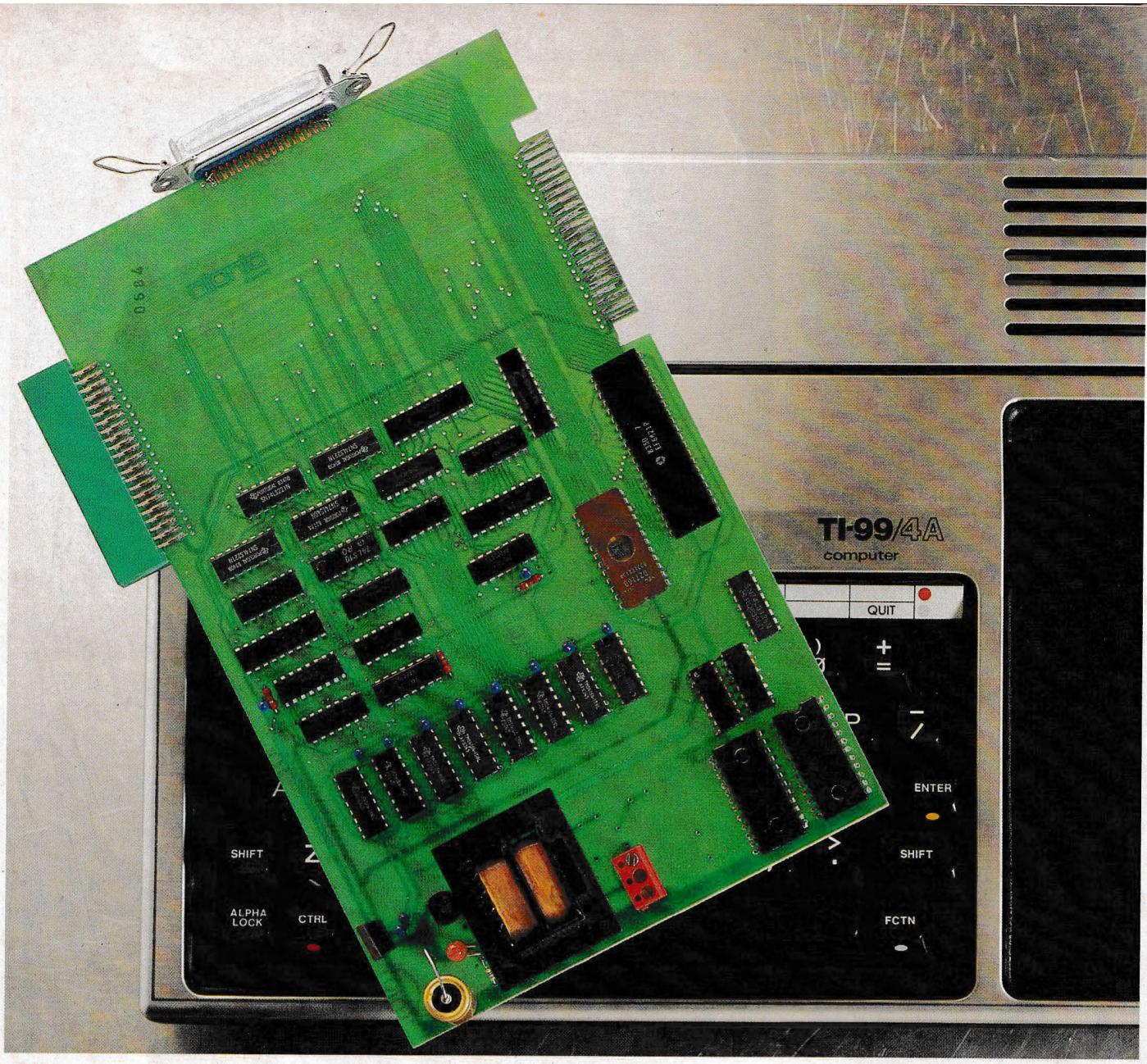
Obwohl die Produktion des Computers und der Peripherie von Texas Instruments eingestellt wurde, gewährleistet TI den Service noch für die nächsten Jahre (wie lange noch), wie uns Olaf Vernunft versicherte. Wendet sich der User an die deutsche Vertretung von TI in Freising bei München, so erhält er dort den Hinweis, daß Erweiterungen für den 99/4A von den Firmen atronic und Klein (in Wuppertal) geliefert werden. Durch diese Hinweise dürfte die Nachfrage nach atronic Systemteilen gesichert sein. Allerdings ergaben sich in den ersten Wochen des Jahres Schwierigkeiten, die Nachfragen der Einzelhändler zu befriedigen. Denn durch den Produktionsstop bei TI stieg die Nachfrage so, daß man nicht in der Lage war, allen Bestellungen sofort nachzukommen. Um solchen Verzögerungen in Zukunft entgegenzutreten, und wegen allgemeiner Expansionspläne, ist man auf die Suche nach einer neuen Bleibe gegangen, in der dann Herstellung und Vertrieb an einem Ort zusammengezogen werden können. Wenn geeignete Voraussetzungen geschaffen sind, wird man sich auch wieder verstärkt der Industrieproduktion zuwenden können.

Als Beispiele für die zu erwartende Breite der Produktpalette seien hier naheliegende Dinge wie Drehzahl-Regeleinheiten für Kunststoffspritzmaschinen und Leistungsverstärker für Diskotheken genannt.

Ganz im Sinne von Forschungsminister Riesenhuber stellen die Jungunternehmer erfreut fest, daß vor allem junge Leute zunehmend Interesse an der Computertechnik zeigen und sich das Image der Branche (wieder) verbessert. Mit hochwertigen Produkten, viel Enthusiasmus und qualifiziertem Personal hoffen sie, atronic den Weg in eine gesicherte Zukunft vorprogrammieren zu können - das Programm TI-Peripherie scheint dazu das richtige zu sein. ■

Peripherie im Test: ZX-81 Speichererweiterung (links) als Bausatz; TI-Peripherie





Speicher im Baukastensystem

Volle Speicher sind gut für magere Jahre. Aber wer viele Ideen hat, der braucht viel Speicherplatz. COMPUTER Praxis testet das Angebot der Firma atronic.

Jeder TI-User kennt das: Man hat eine Idee — Man fängt an zu programmieren, die ersten Randbedingungen tauchen auf; sie werden abgefangen — Neue werden deutlich, auch sie werden gesichert. Und mitten im schönsten Schwung kommt dann die mehr als unerwünschte Meldung MEMORY FULL, Speicher voll. Damit ist unser schönes Programm ersteinmal hin. Und dann geht das Streichen los ...

Es gibt nur einen Ersatz für Speicher: mehr Speicher! Aus diesem Grund haben wir uns umgesehen, welche Speichererweiterungen auf dem Markt erhältlich sind. Der Erfolg: „Ja, wie haben noch eine oder zwei 32K-Karten, aber keine Boxen mehr." — „Ja, wir planen, aber das wird noch eine Weile dauern." — „Ja, zur Zeit bieten wir die 32K- und die 32K-CMOS-Speichererweiterung an, die 64K geht dieser Tage in die Produktion."

Das war vor etwa einem Monat. Die TI-Expansion-Box und die 32K-Karten sind unterdessen überhaupt nicht mehr zu bekommen, aber die 32K-, 32K-CMOS- und die 64K-Speichererweiterungen liegen uns vor. Die Firma atronic aus Hamburg bietet für den TI 99/4A zwei verschiedene Speichererweiterungen in zehn Versionen an, die direkt an den Bus des Rechners angeschlossen werden (siehe unsere Tabelle).

Alle Geräte präsentieren sich im schlichten atronic-anthrazit/alufarbenen Metallgehäuse, Maße B15cm x H8cm x T25cm. Und die neue Serie soll sogar in der Höhe dem TI 99 angeglichen werden, al-

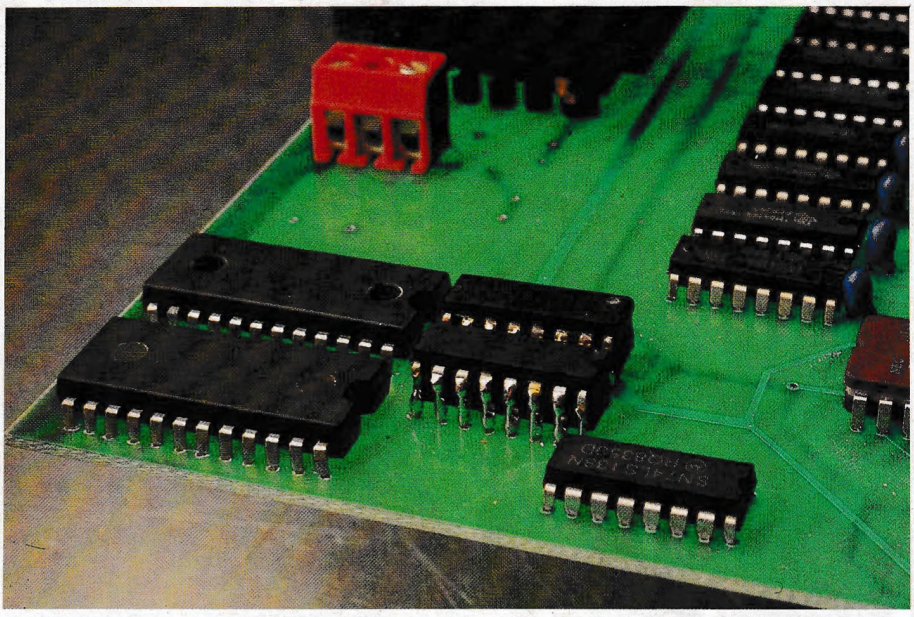
Dynamische Speichererweiterungen

32KByte RAM einfach, oder
mit Extended-BASIC
mit Centronics-Schnittstelle
mit Extended-BASIC und Centronics

64KByte RAM einfach, oder
mit Extended-BASIC
mit Centronics-Schnittstelle
mit Extended-BASIC und Centronics

CMOS-Speicher-erweiterungen

32K-CMOS
ohne Batteriestützung
mit Batteriestützung



Sinnvoll integriert und mit einem Flachkabel an den Modembus anzuschließen: TI-Extended BASIC.

so etwas niedriger sein. Uns wurden die 64K mit Extended-BASIC und Centronics-Schnittstelle sowie die 32K-CMOS vorgelegt.

Was ist RAM? RAM ist eine Abkürzung: Random Access Memory, wörtlich 'Wahlfreier Zugriffs-Speicher' übersetzt, was heißen soll, daß man in diesem Speicher sowohl Programm-Schritte als auch Daten frei ablegen und natürlich auch wieder abfragen kann. Die Adressen der Daten sind dem — Rechner, selten dem User, bekannt.

Was ist KByte? Da muß man mit dem Bit anfangen, also, was ist Bit? Bit ist, wie sollte es anders sein, eine Abkürzung, und zwar für 'binary digit', Binärzahl. Mit einer Binärzahl können zwei Zustände beschrieben werden, 0 und 1. Wollen wir mehr, müssen mehrere Bits, nämlich 8, in einem Byte zusammengefaßt werden. Das Bit ist also der Buchstabe des Binärwortes Byte. Und wenn man 1024 Byte hat, so spricht man von einem KByte. Diese verrückte Zahl hängt mit den zwei Zuständen eines Bits zusammen. Will man mit Bit und Byte rechnen, muß man die Zweier-Potenzen benutzen, und 2^{10} ist 1024. Faule Schreiberlinge kürzen diese Abkürzung dann noch einmal ab, die schreiben nämlich 32K-Speicher. Und das bedeutet, daß eine Speichererweiterung mit 32 x 1024 Bytes = 32.768 Binärwörtern vorliegt. Das ist eine ganze Menge!

Und was ist CMOS? Noch eine Abkürzung: 'Capacity Metal On Silicium'. Wir würden empfehlen, daß sich die interessierten Leser unter euch ein gutes Fachbuch über Elektronik greifen und sich selbst informieren. Hier nur soviel: Bei CMOS handelt es sich nicht mehr um bipolare, sondern um unipolare Transistortechnik. Der Vorteil liegt in dem sehr viel niedrigeren Stromverbrauch. Ansonsten ist ein CMOS-RAM genauso einsetzbar wie ein normales RAM.

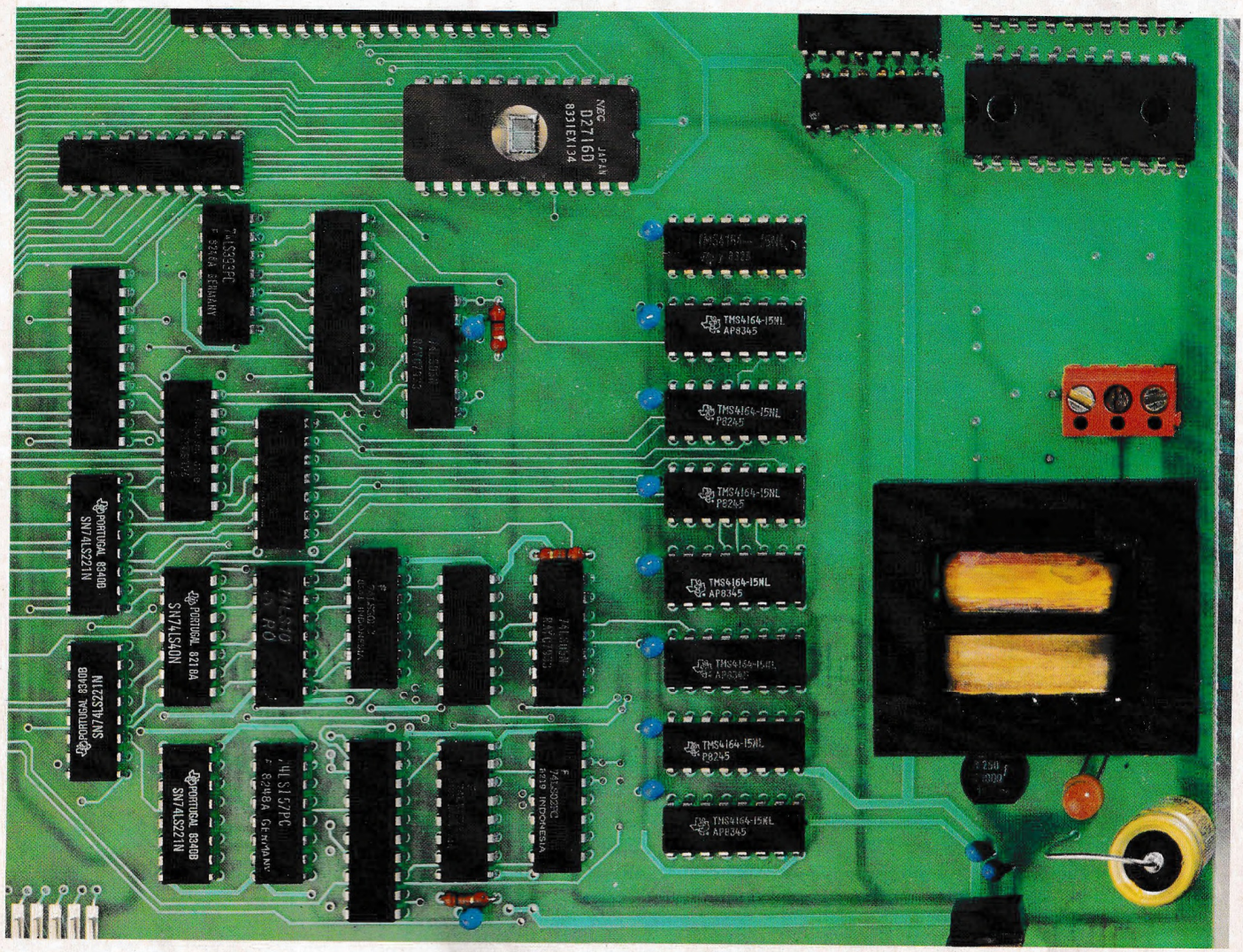
Bleibt noch ein Begriff zu erklären: Centronics. Es handelt sich hierbei um eine Schnittstellen-Normung. Eine Centronics-Schnittstelle ist ein paralleler Port, meist für Drucker genutzt, aber durchaus auch für andere Peripherien verwendbar.

Zurück zu unseren Speichererweiterungen. Wie oben aufgelistet, gibt es die beiden dynamischen RAM-Versionen in jeweils vier Ausführungen: einfach, mit Extended-BASIC, mit Centronics-Schnittstelle und mit beidem.

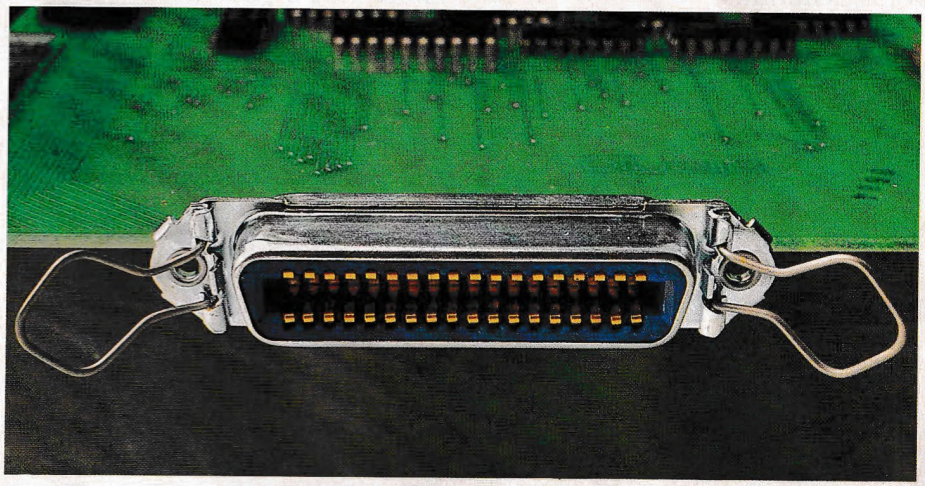
Unser erster Blick galt dem Extended-BASIC. Wir sind sicher, daß eine Welle der Erleichterung durch das Land schwappt, gibt es endlich wieder die Spracherweiterung, die für den TI 99 so überaus wichtig ist. Schon beim Einschalten und Wählen des Extended-BASIC wird klar: Es handelt sich um die original Texas Instruments Software, die hier von atronic eingebaut wurde. Über ein Flachkabel wird die Speichererweiterung an dem Modul-Schacht am TI angeschlossen. Die Betriebssicherheit ist nicht geringer als wenn ein Modul benutzt wird.

Dem Drucker zugewandt

Unser zweiter Blick richtete sich auf die Centronics-Schnittstelle. Auch hier die, jetzt schon erwartete, Betriebssicherheit. Sie entspricht in allen Einzelheiten der bekannten TI-Schnittstelle, die wir in Ausgabe 4/84 „Schnittstellen: maßgeschneidert oder ausbaufähig" beschrieben haben. Es handelt sich hierbei um eine 'maßgeschneiderte' Druckerschnittstelle. Natürlich ist sie mit



oben: Ein Blick auf 64K RAM, Netzteil und steuerndes EPROM.
rechts: Hallo, Drucker!



einer eigenen Stromversorgung ausgerüstet, so daß der Rechner selber nicht belastet wird. Die Schnittstelle ist an der Rückseite des Gerätes aus dem Gehäuse mit einer Centronics-Norm-Buchse herausgeführt. Kabel, um die Peripherie anzuschließen, werden im Fachhandel angeboten.

Nicht zuletzt interessierte uns die Speichererweiterung selber — deswegen haben wir die Geräte schließlich besorgt! 64KByte — das sind 65.536 Speicherworte Kapazität. Eine Erweiterung mit diesem Umfang gab es bisher noch nicht für den TI (jedenfalls nicht in Deutschland). Die Firma atronic hat damit etwas ganz und gar Neues für den TI99/4A auf den Markt gebracht.

Der Unterschied zwischen der 32K- und der 64K-Erweiterung ist, neben der offensichtlichen Kapazität, daß die 32K vollständig für BASIC-Programme nutzbar ist. Das ist bei der 64K leider nicht der Fall. Die große Erweiterung ist nur zur Hälfte für BASIC nutzbar,

die andere Hälfte ist für Daten beziehungsweise Assembler-Programme reserviert.
Wer also nur BASIC programmieren möchte, und auch in Zukunft auf Assembler verzichten will, wird wohl mit der 32K-Erweiterung auskommen. Wem auch 64K zu wenig sind: Dank des durchgeschleiften BUS kann eine weitere 32K-Erweiterung angeschlossen werden. Über eine entsprechende Adressierungs-Routine (Assembler) wird dieser Bereich nutzbar. Der externe Speicher wird somit auf 96K erweitert, eine Möglichkeit, die auch erfahrere Programmierer zufriedenstellt.
CMOS-RAM verbrauchen weniger Strom als dynamische RAM. Diesen Vorteil der Technik macht sich die 32K-CMOS-Erweiterung mit Batteriestützung zunutze. Ein Programm, das in der Erweiterung gespeichert ist, wird bei Abschalten des Rechners nicht gelöscht. Die Batterien sorgen dafür, daß das Programm bis zu einem Jahr erhalten bleibt. Dann sind die Batterien am Ende. Eine tolle Möglichkeit, um häufig gebrauchte Routinen nicht immer erst stundenlang laden zu müssen.

Allerhöchste Flexibilität

Und noch ein Gag, den atronic in die CMOS-Erweiterungen eingebaut hat: die CMOS-ICs stehen auf Sockeln. Der User kann sie herausnehmen und statt dessen eigene EPROMS mit Assembler-oder BASIC-Routinen einstecken.
Die Möglichkeiten, die diese Erweiterungen bieten, sind nahezu unbegrenzt. Was bleibt, ist die Schwierigkeit, zu entscheiden, welche der zehn Versionen für einen die richtige ist. Die Preise sind zwar nicht niedrig, aber durchaus angemessen. Alles, was billiger ist, muß zwangsläufig auf Betriebssicherheit und gute Verarbeitung verzichten. Wobei natürlich die gute Verarbeitung ebenfalls die Betriebssicherheit und somit die Datensicherheit erhöht. Die Preise selbst liegen für die CMOS bei 520,— bzw. 640,— Mark, für die dynamischen RAMs je nach Ausführung zwischen 450,— und 850,— Mark.

Alle Erweiterungen sind zu beziehen von der Firma
atronic
Bunke & Vernunft oHG
Postfach 70 01 24
2000 Hamburg 70
Tel: 040 / 668 22 56 ■

Unser Überlebenskonzept:

O & MD Hamburg

Gewaltfreie Aktionen in aller Welt zur Durchsetzung folgender Ziele:
- Beendigung aller Atombombentests
- Stop der Versenkung von Chemie- und Atommüll im Meer
- Erhaltung der Robben, Wale und Delphine
- Stop dem Sauren Regen und der weiteren Vergiftung unserer Umwelt durch Schadstoffe und Abgase

Ich will das Greenpeace-Überlebenskonzept für unsere Umwelt kennenlernen. Senden Sie mir ausführliches Informationsmaterial.

Name/Vorname

Straße/Nummer

PLZ/Wohnort A 161

Sie zu informieren kostet uns DM 2,40. Schicken Sie diesen Betrag in Briefmarken (falls mehr, bitte Scheck) zusammen mit diesem Coupon an:

GREENPEACE
Greenpeace - Informationsdienst
Hohe Brücke 1 - Haus der Seefahrt - 2000 Hamburg 11
Telefon: 040/37 33 44 oder 37 33 59

Joystick-Adapter selbstgestrickt

Original-Joysticks für den TI sind nicht mehr zu kriegen. Mit etwas Geschick könnt ihr euch selbst einen Adapter bauen und dann systemfremde Joysticks benutzen.

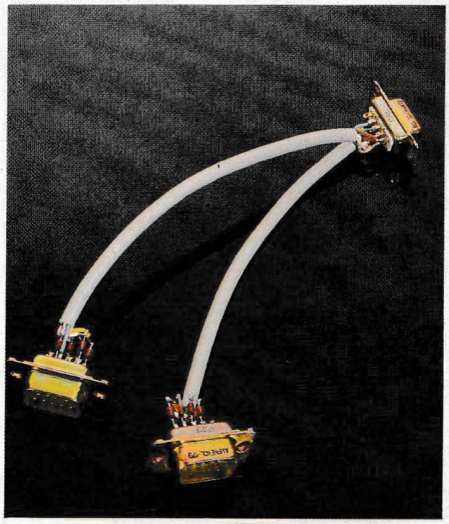
Viele von euch werden Spiele gekauft oder geschrieben haben, die Joysticks benötigen oder die mit Joysticks mehr Spaß machen. Vor allem, wenn man zu zweit spielt, ist eine Tastatur arg eng.

Viel schöner ist das Spiel doch mit einem Paar gut in der Hand liegender, leichtgängiger und betriebssicherer Joysticks. Darum haben wir uns hier in der Redaktion für unsere Software-Tests auch welche zugelegt. Blieb nur noch das Problem, ob wir für den TI 99/4A die TI-Joysticks kaufen sollten — oder nicht. Auf dem Markt sind Adapter erhältlich, die es erlauben, Atari- oder Atarikompatible Joysticks zu benutzen. Diese Adapter sind nicht ganz billig (ab ca. DM 30,—). Wer aber die TI-Joysticks kennt, weiß, daß die Dinger einem nach kurzer Zeit das Spiel verleiden können. Also doch den Adapter.

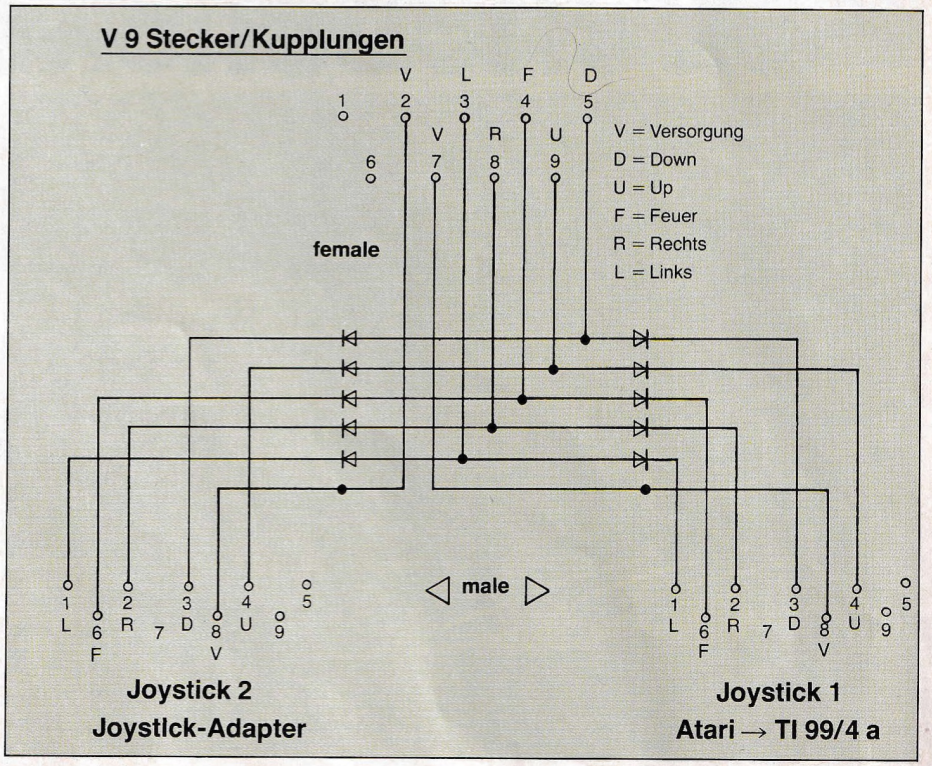
Wir wären eine sonderbare Redaktion für eine Computer-Zeitschrift, wenn wir nicht sofort zu Schraubendreher und Lötkolben gegriffen hätten: „Sowas muß man doch selbst bauen können ...“ — Und tatsächlich, mit minimalem Aufwand hatten wir einen Adapter gebastelt. Wir haben dafür gebraucht:

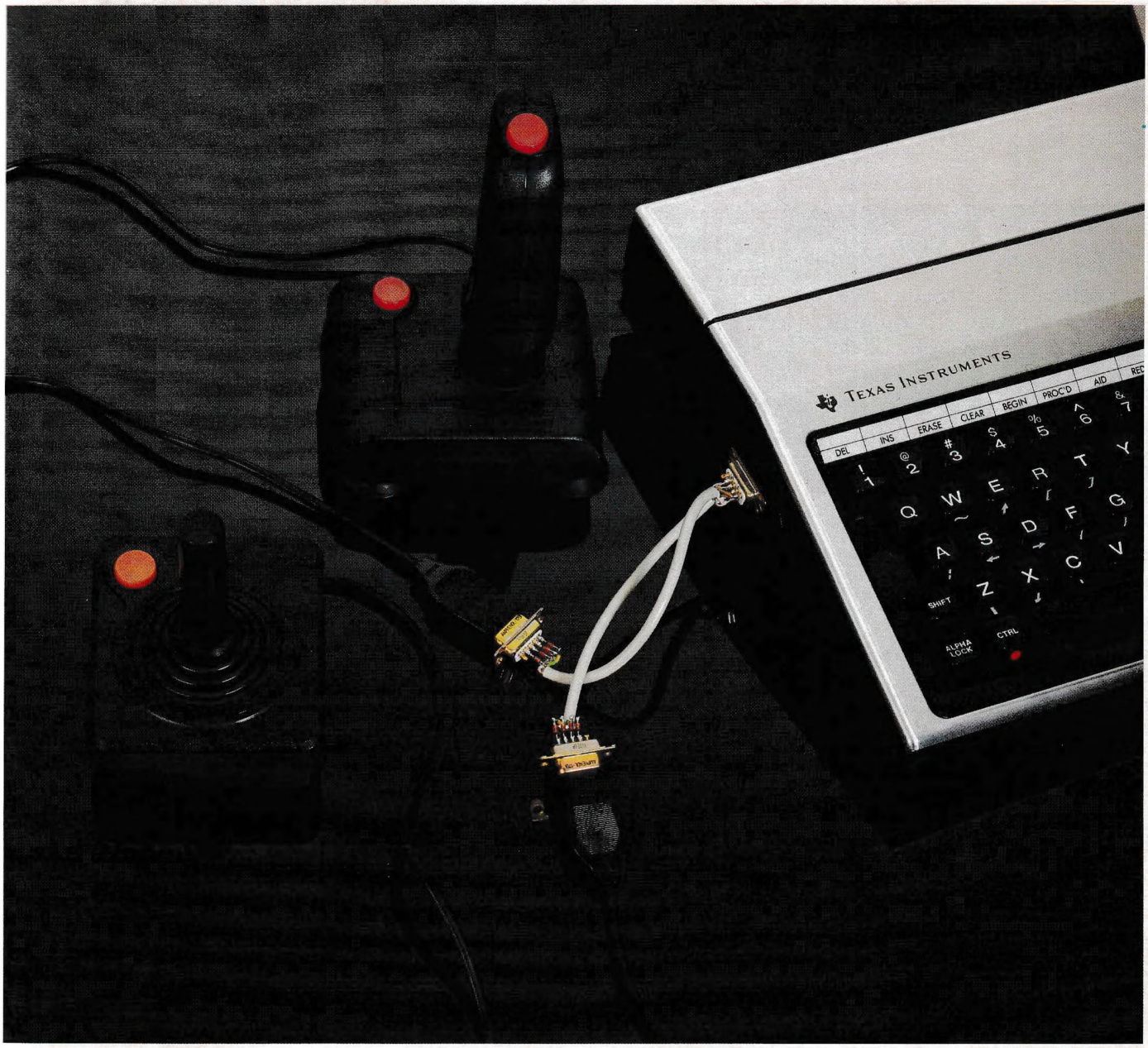
- ▷ eine 9-Pol-Kupplung
- ▷ zwei 9-Pol-Stecker
- ▷ 10 Dioden

und natürlich Kabel. Die Pins der Stecker und der Kupplung sind numeriert, die Dioden sind gekennzeichnet, so daß der Zusammenbau keine Probleme bereitet: Die Dioden werden einfach in die Stecker für die Joysticks gelötet. Der ganze Adapter kostet etwa 15,— Mark, alle Bauteile sind in Elektronik-Fachgeschäften normalerweise erhältlich.



Für die Bastler und Sparsamen unter den Spielfreunden: das Schaltbild zu unserem Joystick-Adapter. Ihr werdet feststellen: Er ist tatsächlich so einfach nachzubauen, wie er aussieht.





TI 99/4A ***Erweitern Sie Ihr System*** **TI 99/4A**

Nutzen Sie den Telefonkontakt und fordern Sie Unterlagen an: **040 - 668 22 56**

- *volle Kompatibilität zum TI 99/4A*
- *softwaremässige Anpassung*
- *alle Komponenten einzeln funktionsfähig*
- *Erweiterung durch einfaches Aneinanderstecken*

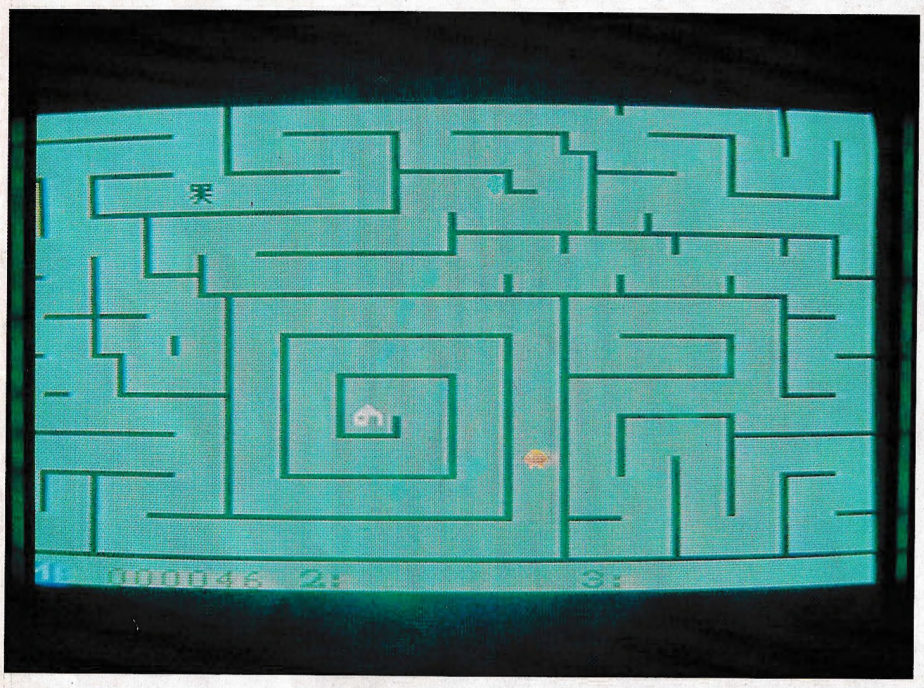
mit System!

Schnittstellen - Speichererweiterungen - Disketten-Steuerungen - Disk-Laufwerke

atronic Bunke & Vernunft oHG
Postfach 700 124
2000 Hamburg 70

SCHWEINCHENSUCHE

Sauhatz in BASIC und X-BASIC



Es handelt sich bei diesem Spiel um ein Labyrinthspiel mit der Sonderaufgabe, auf dem Weg zum Ziel möglichst viele der unterwegs zufällig auftauchenden Schweinchen einzusammeln. Der Spieler muß aber ein Risiko bedenken: wenn sein Männchen erst einmal losgelaufen ist, wird es nur noch von den Wänden, einem Schweinchen oder dem Haus aufgehalten. Läuft es gegen eine der Wände des Labyrinths, werden Punkte abgezogen. Trifft es jedoch auf ein Schweinchen oder läuft in das Haus, gibt es Plus-Punkte. Wenn das Männchen das Haus erreicht hat, kommt der nächste Spieler bzw. die nächste von insgesamt drei Runden.

Zum Listing:

- ▷ die Zeilen 100-290 definieren den Zeichensatz
- ▷ in den Zeilen 300-690 wird das Spiel vorbereitet, die Anzahl der Spieler (1-3) und die Lauf-Geschwindigkeit des Männchens festgelegt
- ▷ in den Zeilen 700-2730 wird das Labyrinth aufgebaut. Wer Spaß daran hat, kann hier eigene Gestaltungs-Ideen für das Labyrinth verwirklichen
- ▷ in den Zeilen 2740-3030 wird festgestellt, welche Runde und welcher Spieler dran ist
- ▷ die Zeilen 3040-3160 suchen die Schweinchen und machen ihre Position im Labyrinth sichtbar
- ▷ wer zu oft gegen die Wände gelaufen ist, wird durch die Zeilen 3170-3230 rausgeworfen
- ▷ die Zeilen 3240-4190 steuern das Männchen, prüfen, ob es gegen eine Wand, auf ein Schwein getroffen oder in das Haus gelaufen ist
- ▷ in den Zeilen 4200-4390 prüft das Programm, welcher Spieler als nächster an der Reihe ist
- ▷ von Zeile 4400 bis Zeile 4770 werden die End-Punktzahlen ermittelt und ausgegeben. Sie errechnen sich jeweils aus der Geschwindigkeit des Männchens, der erzielten Punkte und, als Bonus, der Anzahl der eingesammelten Schweinchen
- ▷ die Zeilen 4780-5050 beinhalten drei Sub-Routinen:
  — 4780-4880 schreibt die Punkte während des Spiels.
  — 4890-4950 berechnet den Punktabzug, wenn das Männchen gegen eine Wand gelaufen ist.
  — 4960-5050 addiert die Punkte für ein eingesammeltes Schweinchen.

Das hier abgedruckte Listing ist für die Spiel-Steuerung über Tastatur ausgelegt. Um mit einem Joystick spielen zu können, sind nur die folgenden acht Änderungen erforderlich:

Zeile	Neuer Befehl
3000	CALL KEY (1, K, ST)
3250	CALL KEY (1, K, ST)
3300	CALL JOYST (1, X, Y)
3310	IF X > 0 THEN 3360
3320	IF X < 0 THEN 3760
3330	IF Y > 0 THEN 3980
3340	IF Y < 0 THEN 3580
4720	CALL KEY (1, K, ST)

Diese Änderungen ermöglichen die Steuerung der Funktionen mit Joystick 1.

. . . und nun geht's los

Wenn ihr das Spiel startet, werden zunächst die Zeichen definiert. Dann erscheint das Startbild mit der Frage nach der Anzahl der Spieler (1-3 Spieler sind möglich). Habt ihr korrekte Eingaben gemacht, fragt das Programm nach der gewünschten Geschwindigkeit. Hier werden nur die Werte 1-5 akzeptiert (1 steht für die niedrigste, 5 für die höchste Geschwindigkeit). Ist auch dieser

Wert korrekt eingegeben worden, wird das Labyrinth aufgebaut. In der linken oberen Ecke steht das Männchen, in der Feldmitte das Haus. Darüber wird angezeigt, welcher Spieler an der Reihe ist und welche Runde gerade gespielt wird. Ihr müßt jetzt irgendeine Taste (bzw. die Fire-Taste) drükken, dann taucht irgendwo das erste Schweinchen auf. Auf erneuten Tastendruck (bzw. Fire) läuft das Männchen los. Mit den Tasten E, S, D, X wird die Richtung geändert, in die das Männchen läuft. Trifft es auf ein Schweinchen, werden die Plus-Punkte addiert, angezeigt, und es erscheint so lange ein neues Schweinchen, bis alle im Spiel befindlichen eingesammelt sind. Man kann das Männchen aber auch direkt ins Haus gehen lassen, ohne unterwegs Schweinchen einzusammeln. Das Männchen verschwindet dann, der nächste Spieler bzw. die nächste Runde wird aufgerufen und das Männchen erscheint wieder in der linken oberen Ecke. Die folgenden Runden werden genauso gesteuert wie die erste.

In jeder Runde gibt es für jeden Spieler neue Schweinchen, die es einzusammeln gilt. Aus der Vorrunde übriggebliebene Schweinchen bleiben stehen und können vom nächsten Spieler eingesammelt werden (was seinem Punkte-Konto zugute kommt!)

Nach der dritten Runde werden die Gesamtpunktzahlen und der Highscore ausgegeben. Die Gesamtpunktzahl errechnet sich aus der Geschwindigkeit, den erspielten Punkten und der Anzahl der eingesammelten Schweinchen.

In der oberen Hälfte des Bildschirms blinken zwei Zeilen:

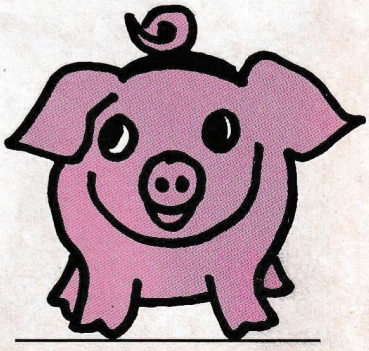
Sieger: X
?

X steht für den Sieger des Spiels, das Fragezeichen für die Frage, ob ihr noch einmal spielen wollt. Wenn ja, müßt ihr irgendeine (bzw. die Fire-) Taste drücken und halten. Nach einer Weile wird der Bildschirm gelöscht und das Programm fährt mit der Frage: ‚Wieviele Spieler?' (1-3) fort.

Noch eine Bemerkung zur Eingabe des Listings: Alle Kommentare müssen mit Kleinbuchstaben geschrieben werden, da die ASCII-Zeichen der Großbuchstaben umdefiniert sind. Das Programm läuft schneller in TI-XBASIC.

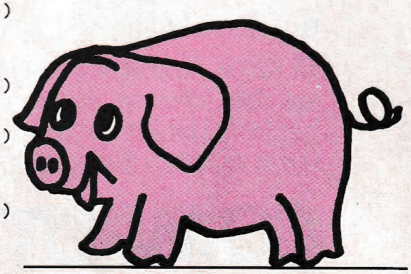
SCHWEINCHENSUCHE

```
100 RANDOMIZE
110 REM CHAR-DEFINITION
120 DATA "000000FFFF","1818181818181818",
    "000000F8F8181818","0000001F1F181818",
    "181818F8F8","1818181F1F"
130 DATA "181818FFFF181818","7E5A7E183E581422",
    "183878F8F8783818","181C1E1F1F1E1C18",
    "000000FFFF7E3C18"
140 DATA "183C7EFFFF","000000F0F0","0000000F0F",
    "18181818","0000000018181818","000000FFFF181818",
    "181818FFFF"
150 DATA "181818F8F8181818","1818181F1F181818"
160 FOR I=34 TO 47
170 READ A$
180 CALL CHAR(I,A$)
190 NEXT I
200 FOR I=64 TO 69
210 READ A$
220 CALL CHAR(I,A$)
230 NEXT I
240 CALL CHAR(72,"00183C5EFFFE3C24")
250 CALL CHAR(80,"183C7EFF9191F1F1")
260 CALL CHAR(106,"0C1C3C6C0C0C0C0C")
270 CALL CHAR(113,"3C6242060C3B7F")
280 CALL CHAR(120,"3C66021C02663C")
290 CALL CHAR(126,"3C7EC3C3660C000C")
300 REM SPIELVORBEREITUNG
310 DIM SP(3,2)
320 CALL CHAR(33,"3C4299A1A199423C")
330 CALL CLEAR
340 PRINT "schweinchensuche": :" ! eumel 1983":
350 PRINT :"highscore : ";PM: : : :
360 INPUT "wieviel spieler? )1 bis 3(: ":SLR
370 IF (SLR<1)+(SLR>3)THEN 360
380 PRINT
390 INPUT "tempo )1 bis 5(: ":H
400 PRINT
410 IF (H<1)+(H>5)THEN 390
420 ON H GOTO 430,450,470,490,510
430 HTE=60
440 GOTO 520
450 HTE=50
460 GOTO 520
470 HTE=35
480 GOTO 520
490 HTE=15
500 GOTO 520
510 HTE=1
520 FOR I=1 TO 3
530 SP(I,1)=0
540 SP(I,2)=0
550 NEXT I
560 F=1
570 BL=0
580 CALL COLOR(1,2,4)
590 CALL COLOR(2,2,4)
600 CALL COLOR(5,2,4)
610 CALL COLOR(3,7,4)
620 CALL COLOR(4,7,4)
630 CALL COLOR(6,7,4)
640 CALL COLOR(7,16,4)
650 FOR I=9 TO 12
660 CALL COLOR(I,13,4)
670 NEXT I
```



```
680 CALL SCREEN(2)
690 CALL CLEAR
700 REM AUFBAU
710 CALL HCHAR(2,1,34,6)
720 CALL HCHAR(2,7,46)
730 CALL VCHAR(1,9,35,3)
740 CALL HCHAR(4,1,34)
750 CALL HCHAR(4,2,36)
760 CALL VCHAR(5,2,35,2)
770 CALL HCHAR(7,2,64)
780 CALL HCHAR(4,4,37)
790 CALL HCHAR(4,5,34,2)
800 CALL HCHAR(4,7,46)
810 CALL VCHAR(5,4,35)
820 CALL HCHAR(6,4,39)
830 CALL HCHAR(6,5,34)
840 CALL HCHAR(6,6,66)
850 CALL HCHAR(6,7,34,8)
860 CALL HCHAR(6,15,38)
870 CALL VCHAR(5,15,35)
880 CALL VCHAR(4,15,69)
890 CALL VCHAR(3,15,35)
900 CALL VCHAR(2,15,36)
910 CALL HCHAR(2,11,47)
920 CALL HCHAR(2,12,34,3)
930 CALL HCHAR(4,9,39)
940 CALL HCHAR(4,10,34,3)
950 CALL HCHAR(4,13,46)
960 CALL HCHAR(4,16,34,3)
970 CALL HCHAR(4,19,36)
980 CALL HCHAR(5,19,39)
990 CALL HCHAR(5,20,46)
1000 CALL VCHAR(7,6,35,2)
1010 CALL VCHAR(9,6,39)
1020 CALL HCHAR(9,7,34)
1030 CALL VCHAR(9,8,68)
1040 CALL VCHAR(8,8,65)
1050 CALL VCHAR(10,8,39)
1060 CALL HCHAR(8,10,47)
1070 CALL HCHAR(8,11,34,6)
1080 CALL HCHAR(8,17,38)
1090 CALL HCHAR(7,17,69)
1100 CALL VCHAR(6,17,65)
1110 CALL HCHAR(7,18,34,3)
1120 CALL HCHAR(7,21,66)
1130 CALL VCHAR(8,21,64)
1140 CALL HCHAR(7,22,67)
1150 CALL VCHAR(5,22,35,2)
1160 CALL VCHAR(4,22,69)
1170 CALL HCHAR(4,23,34,2)
1180 CALL HCHAR(4,25,46)
1190 CALL HCHAR(3,22,36)
1200 CALL HCHAR(3,21,39)
1210 CALL HCHAR(2,21,36)
1220 CALL HCHAR(2,17,47)
1230 CALL HCHAR(2,18,34,3)
1240 CALL HCHAR(7,23,34)
1250 CALL HCHAR(7,24,67)
1260 CALL VCHAR(6,24,65)
1270 CALL HCHAR(7,25,66)
1280 CALL VCHAR(8,25,64)
1290 CALL HCHAR(7,26,34,3)
1300 CALL HCHAR(7,29,66)
1310 CALL VCHAR(8,29,64)
1320 CALL HCHAR(7,30,34)
1330 CALL HCHAR(7,31,36)
1340 CALL VCHAR(8,31,35)
1350 CALL VCHAR(9,31,39)
1360 CALL HCHAR(9,32,34)
1370 CALL VCHAR(1,23,64)
1380 CALL HCHAR(2,24,47)
1390 CALL HCHAR(2,25,34,2)
1400 CALL HCHAR(2,27,36)
1410 CALL VCHAR(3,27,35,2)
1420 CALL VCHAR(1,29,35,2)
1430 CALL VCHAR(3,29,64)
1440 CALL HCHAR(5,27,39)
1450 CALL HCHAR(5,28,34,3)
1460 CALL HCHAR(5,31,38)
1470 CALL VCHAR(3,31,35,2)
1480 CALL VCHAR(2,31,65)
1490 CALL HCHAR(9,1,34)
1500 CALL HCHAR(9,2,46)
1510 CALL VCHAR(8,4,65)
1520 CALL VCHAR(9,4,35,2)
1530 CALL HCHAR(11,4,40)
1540 CALL HCHAR(11,3,34)
1550 CALL HCHAR(11,2,47)
1560 CALL HCHAR(11,5,34)
1570 CALL HCHAR(11,6,46)
1580 CALL VCHAR(12,4,35)
1590 CALL HCHAR(13,4,39)
1600 CALL HCHAR(13,5,36)
1610 CALL VCHAR(14,5,35)
1620 CALL HCHAR(15,5,39)
1630 CALL HCHAR(15,6,34,3)
1640 CALL VCHAR(12,7,65)
1650 CALL VCHAR(13,7,64)
1660 CALL HCHAR(15,2,47)
1670 CALL HCHAR(15,3,46)
1680 CALL HCHAR(13,1,34)
1690 CALL HCHAR(13,2,46)
1700 CALL HCHAR(17,1,34,6)
1710 CALL HCHAR(17,7,46)
1720 CALL VCHAR(21,2,64)
1730 CALL VCHAR(20,2,35)
1740 CALL VCHAR(19,2,37)
1750 CALL HCHAR(19,3,34)
1760 CALL HCHAR(19,4,66)
1770 CALL HCHAR(19,5,34,2)
1780 CALL HCHAR(19,7,46)
1790 CALL VCHAR(20,4,35,3)
1800 CALL HCHAR(21,6,47)
1810 CALL HCHAR(21,7,34,2)
1820 CALL VCHAR(9,19,65)
1830 CALL VCHAR(9,23,65)
1840 CALL HCHAR(10,23,67)
1850 CALL HCHAR(10,22,34)
1860 CALL HCHAR(10,24,34,3)
1870 CALL HCHAR(10,27,67)
1880 CALL VCHAR(9,27,65)
1890 CALL HCHAR(10,28,34)
1900 CALL HCHAR(10,29,36)
1910 CALL VCHAR(11,29,69)
1920 CALL HCHAR(11,30,34)
1930 CALL HCHAR(11,31,46)
1940 CALL VCHAR(12,29,35,3)
1950 CALL HCHAR(15,29,39)
1960 CALL HCHAR(15,30,34)
1970 CALL HCHAR(15,31,46)
1980 CALL HCHAR(13,32,34)
1990 CALL HCHAR(13,31,47)
2000 CALL HCHAR(12,23,47)
2010 CALL HCHAR(12,24,34,3)
2020 CALL HCHAR(12,27,36)
2030 CALL VCHAR(13,27,35)
2040 CALL HCHAR(14,27,38)
2050 CALL HCHAR(14,22,34,5)
2060 CALL HCHAR(17,28,34,5)
2070 CALL VCHAR(17,27,69)
2080 CALL VCHAR(18,27,35,3)
2090 CALL VCHAR(21,27,64)
2100 CALL HCHAR(16,27,36)
2110 CALL HCHAR(16,24,34,3)
2120 CALL HCHAR(16,23,37)
2130 CALL VCHAR(17,23,35,2)
2140 CALL VCHAR(19,23,64)
2150 CALL HCHAR(21,22,34)
2160 CALL HCHAR(21,23,46)
2170 CALL VCHAR(18,25,65)
2180 CALL VCHAR(19,25,35,4)
2190 CALL HCHAR(19,31,46)
2200 CALL HCHAR(19,30,34)
2210 CALL HCHAR(19,29,37)
2220 CALL VCHAR(20,29,35,3)
2230 CALL HCHAR(21,32,34)
2240 CALL HCHAR(21,31,47)
2250 CALL HCHAR(23,1,34,3)
2260 CALL HCHAR(23,4,67)
2270 CALL HCHAR(23,5,34,16)
2280 CALL HCHAR(23,21,67)
2290 CALL HCHAR(23,22,34,3)
2300 CALL HCHAR(23,25,67)
2310 CALL HCHAR(23,26,34,3)
2320 CALL HCHAR(23,29,67)
2330 CALL HCHAR(23,30,34,3)
2340 CALL VCHAR(22,21,35)
2350 CALL HCHAR(21,21,69)
2360 CALL VCHAR(15,21,35,6)
2370 CALL VCHAR(14,21,69)
2380 CALL VCHAR(11,21,35,3)
2390 CALL VCHAR(10,21,66)
2400 CALL HCHAR(10,20,34)
2410 CALL HCHAR(10,19,67)
2420 CALL HCHAR(10,10,34,9)
2430 CALL VCHAR(10,9,66)
2440 CALL VCHAR(11,9,35,4)
2450 CALL VCHAR(15,9,68)
2460 CALL VCHAR(16,9,35,5)
2470 CALL VCHAR(21,9,67)
2480 CALL HCHAR(21,10,34,9)
2490 CALL VCHAR(21,19,38)
2500 CALL VCHAR(13,19,35,8)
2510 CALL HCHAR(12,19,36)
2520 CALL HCHAR(12,12,34,7)
2530 CALL VCHAR(12,11,37)
2540 CALL VCHAR(13,11,35,6)
2550 CALL HCHAR(19,11,39)
2560 CALL HCHAR(19,12,34,5)
2570 CALL VCHAR(19,17,38)
2580 CALL VCHAR(15,17,35,4)
2590 CALL HCHAR(14,17,36)
2600 CALL HCHAR(14,14,34,3)
2610 CALL VCHAR(14,13,37)
2620 CALL VCHAR(15,13,35,2)
2630 CALL HCHAR(17,13,39)
2640 CALL HCHAR(17,14,34)
2650 CALL VCHAR(17,15,38)
2660 CALL VCHAR(16,15,65)
2670 CALL HCHAR(24,2,106)
2680 CALL HCHAR(24,3,58)
2690 CALL HCHAR(24,12,113)
2700 CALL HCHAR(24,13,58)
2710 CALL HCHAR(24,22,120)
2720 CALL HCHAR(24,23,58)
2730 CALL HCHAR(1,1,41)
2740 REM SPIELER?
2750 IF F>1 THEN 2780
2760 BL=BL+1
2770 IF BL>3 THEN 4410
2780 RESTORE 2790
2790 DATA 114,117,110,100,
     101,115,112,105,101,
     108,101,114
2800 FOR I=1 TO 5
2810 READ A
2820 CALL HCHAR(13,11+I,A)
2830 NEXT I
2840 CALL HCHAR(13,18,48+BL)
2850 FOR I=1 TO 7
2860 READ A
2870 CALL HCHAR(11,10+I,A)
```

```
2880 NEXT I
2890 CALL HCHAR(11,19,48+F)
2900 CALL HCHAR(16,14,80)
2910 IF SP(F,1)>0 THEN 2930
2920 SP(F,1)=50
2930 SP(F,1)=SP(F,1)+3*SP(F,2)
2940 GOSUB 4790
2950 Z=1
2960 S=1
2970 CALL COLOR(9+F,12,5)
2980 R=1
2990 SCHW=0
3000 CALL KEY(0,K,ST)
3010 IF ST=0 THEN 3000
3020 CALL HCHAR(13,12,32,7)
3030 CALL HCHAR(11,11,32,9)
3040 REM SCHWEIN SET
3050 IF SCHW>13 THEN 3180
3060 ZS=20*RND+1
3070 SS=30*RND+1
3080 CALL GCHAR(ZS,SS,I)
3090 IF I<>32 THEN 3060
3100 CALL HCHAR(ZS,SS,72)
3110 CALL SOUND(100,440,0)
3120 CALL SOUND(100,294,0)
3130 FOR I=0 TO 30 STEP 3
3140 CALL SOUND(-50,392,I)
3150 NEXT I
3160 SCHW=SCHW+1
3170 REM ABSTURZ
3180 IF SP(F,1)>0 THEN 3250
3190 FOR I=120 TO 0 STEP -1
3200 CALL SOUND(-100,110+I,INT(I/8))
3210 NEXT I
3220 CALL HCHAR(Z,S,32)
3230 GOTO 4300
3240 REM START/STEUERUNG
3250 CALL KEY(0,K,ST)
3260 CALL SOUND(5,110,10)
3270 IF ST=0 THEN 3180
3280 FOR I=1 TO HTE
3290 NEXT I
3300 CALL KEY(3,K,ST)
3310 IF K=68 THEN 3360
3320 IF K=83 THEN 3760
3330 IF K=69 THEN 3980
3340 IF K=88 THEN 3580
3350 ON R GOTO 3360,3580,3760,3980
3360 R=1
3370 CALL HCHAR(Z,S,32)
3380 IF S=32 THEN 3540
3390 S=S+1
3400 CALL GCHAR(Z,S,W)
3410 IF W<>32 THEN 3440
3420 CALL HCHAR(Z,S,41)
3430 GOTO 3280
3440 IF W=72 THEN 3510
3450 CALL HCHAR(Z,S,42)
3460 GOSUB 4900
3470 CALL HCHAR(Z,S,W)
3480 S=S-1
3490 CALL HCHAR(Z,S,41)
3500 GOTO 3180
3510 CALL HCHAR(Z,S,41)
3520 GOSUB 4970
3530 GOTO 3050
3540 CALL HCHAR(Z,S,42)
3550 GOSUB 4900
3560 CALL HCHAR(Z,S,41)
3570 GOTO 3180
3580 R=2
3590 CALL HCHAR(Z,S,32)
3600 Z=Z+1
3610 CALL GCHAR(Z,S,W)
```



COMPUTER TCS TELDEC SOFTWARE

Für Commodore 64 und VC20

TIGERVISION™ SEGA

HAYDEN SOFTWARE

Sierra ON-LINE INC

Datasoft Inc.® COMPUTER SOFTWARE

SUPERSPIELE DIE ALLES BRINGEN

Springer VC20 Modul
Mit dem Hasen auf Wolken hüpfen.
Polaris VC20 Modul
Unterwasser-Action aus der Spielhalle.
Jawbreaker VC20 64 Modul
Süßer Spaß in der Bonbon-Fabrik.
Threshold VC20 64 Modul
Weltraum-Abenteuer mit viel Action.
Crossfire VC20 Modul 64 Cassette
Die Stadt muß verteidigt werden – Arkaden-Spiel!
Lunar Leeper VC20 64 Modul
Spannende Abenteuer auf dem Mond – Spitze!
Cannonblitz Ball VC20 Modul
Arkaden-Spiel – mit vielen Stufen – wer erreicht die Flagge der Rotröcke?
Creepy Corridors VC20 Modul
Juwelen-Jagd im Labyrinth – doch da lauern Gefahren!
Ultima Escape VC20 Cassette (8K)
Ganz aktuelles Grafik-Adventure – jetzt auch für VC20.
Frogger 64 Diskette oder Cassette VC20 Cassette (8K)
Das ganz berühmte Arkadenspiel – als Original-Computer-Version.
Mr. Cool 64 Modul
Ein irre schnelles Kletterspiel – da fällt es schwer, cool zu bleiben.
Sammy Lightfoot 64 Diskette
Ein Spiel mit Spitzen-Grafik – für echte Artisten.
Oils Well 64 Modul
Nach Öl bohren, eine Pipeline bauen und dabei vielen Gefahren ausweichen – ein neuer Spiele-Hit.
Quest for Tires 64 Diskette
Der berühmte Steinzeit-Comic-Strip jetzt als Action-Game – bringt irre Laune.
Buck Rogers VC20 64 Modul
Der Filmhit jetzt als superschnelles Computerspiel.
Star Trek VC20 64 Modul
Geschicklichkeit und Konzentration – die Angriffe der Klingdons sind abzuwehren.
Congo Bongo VC20 64 Modul
Das bekannte Arkaden-Spiel – wer besiegt den Urwald-Gorilla Congo Bongo?
Sands of Egypt 64 Diskette
Ein spannendes Grafik-Adventure Spiel.
Pooyan 64 Cassette Diskette
Wie in der Spielhalle – spannend und lustig zugleich.
O'Rileys Mine 64 Cassette Diskette
Abenteuer im Bergwerk – es geht um Punkte.
Moon Shuttle 64 Cassette
Super-Applikation des bekannten Arkaden-Spieles.
Genesis 64 Cassette Diskette
3-D-Vektor-Grafik und superschnell – wie in der Spielhalle.
Sargon II 64 Cassette Diskette
Bekanntes, spielstarkes Schachprogramm.

Die Programme gibt's im Fachhandel und in den Spezial-Abteilungen der Kaufhäuser. Zu unverbl. empf. Verkaufspreisen zwischen DM 99,– und DM 149,–

Modul = ROM-Modul · VC20 = VC20 · Cassette = Cassette · Diskette = Diskette · 64 = 64

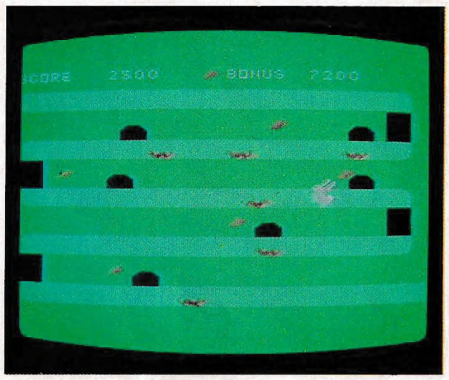
MUSIK+ FREIZEIT SERVICE

TELDEC Musik und Freizeit Service · Heußweg 25 · 2000 Hamburg 19

```
3620 IF W<>32 THEN 3650
3630 CALL HCHAR(Z,S,41)
3640 GOTO 3280
3650 IF W=72 THEN 3730
3660 IF W=80 THEN 4210
3670 CALL HCHAR(Z,S,45)
3680 GOSUB 4900
3690 CALL HCHAR(Z,S,W)
3700 Z=Z-1
3710 CALL HCHAR(Z,S,41)
3720 GOTO 3180
3730 CALL HCHAR(Z,S,41)
3740 GOSUB 4970
3750 GOTO 3050
3760 R=3
3770 CALL HCHAR(Z,S,32)
3780 IF S=1 THEN 3940
3790 S=S-1
3800 CALL GCHAR(Z,S,W)
3810 IF W<>32 THEN 3840
3820 CALL HCHAR(Z,S,41)
3830 GOTO 3280
3840 IF W=72 THEN 3910
3850 CALL HCHAR(Z,S,43)
3860 GOSUB 4900
3870 CALL HCHAR(Z,S,W)
3880 S=S+1
3890 CALL HCHAR(Z,S,41)
3900 GOTO 3180
3910 CALL HCHAR(Z,S,41)
3920 GOSUB 4970
3930 GOTO 3050
3940 CALL HCHAR(Z,S,43)
3950 GOSUB 4900
3960 CALL HCHAR(Z,S,41)
3970 GOTO 3180
3980 R=4
3990 CALL HCHAR(Z,S,32)
4000 IF Z=1 THEN 4160
4010 Z=Z-1
4020 CALL GCHAR(Z,S,W)
4030 IF W<>32 THEN 4060
4040 CALL HCHAR(Z,S,41)
4050 GOTO 3280
4060 IF W=72 THEN 4130
4070 CALL HCHAR(Z,S,44)
4080 GOSUB 4900
4090 CALL HCHAR(Z,S,W)
4100 Z=Z+1
4110 CALL HCHAR(Z,S,41)
4120 GOTO 3180
4130 CALL HCHAR(Z,S,41)
4140 GOSUB 4970
4150 GOTO 3050
4160 CALL HCHAR(Z,S,44)
4170 GOSUB 4900
4180 CALL HCHAR(Z,S,41)
4190 GOTO 3180
4200 REM ENDE RUNDE
4210 CALL HCHAR(Z,S,W)
4220 CALL SOUND(100,262,0,330,0,392,0)
4230 CALL SOUND(100,330,0,392,0,523,0)
4240 CALL SOUND(100,392,0,523,0,659,0)
4250 FOR I=0 TO 30
4260 CALL SOUND(-50,523,I,659,I,784,I)
4270 NEXT I
4280 SP(F,1)=SP(F,1)+153
4290 GOSUB 4790
4300 CALL COLOR(9+F,13,4)
4310 IF SLR=1 THEN 2730
4320 ON F GOTO 4330,4350,4380
4330 F=2
4340 GOTO 2730
4350 IF SLR=2 THEN 4380
4360 F=3
4370 GOTO 2730
4380 F=1
4390 GOTO 2730
4400 REM GAME OVER
4410 CALL CLEAR
4420 PRINT "* * * *  game over  * * * *  "
4430 FOR I=1 TO 3
4440 SP(I,1)=SP(I,2)*SP(I,1)*H*3
4450 NEXT I
4460 CALL SCREEN(4)
4470 PRINT "punkte","schweinchen H": :
     "j :";SP(1,1),SP(1,2):"q :";SP(2,1),SP(2,2):
     "x :";SP(3,1),SP(3,2):
4480 IF SP(1,1)>SP(2,1)THEN 4540
4490 IF SP(2,1)>SP(3,1)THEN 4520
4500 SG=120
4510 GOTO 4560
4520 SG=113
4530 GOTO 4560
4540 IF SP(1,1)<SP(3,1)THEN 4500
4550 SG=106
4560 RESTORE 4570
4570 DATA 115,105,101,103,101,114,32,58,32
4580 IF PM>SP(1,1)THEN 4600
4590 PM=SP(1,1)
4600 IF PM>SP(2,1)THEN 4620
4610 PM=SP(2,1)
4620 IF PM>SP(3,1)THEN 4640
4630 PM=SP(3,1)
4640 PRINT :"highscore:";PM
4650 CALL HCHAR(5,16,126)
4660 FOR I=1 TO 9
4670 READ A
4680 CALL HCHAR(3,11+I,A)
4690 NEXT I
4700 CALL HCHAR(3,21,SG)
4710 RESTORE 4570
4720 CALL KEY(0,K,ST)
4730 CALL HCHAR(5,16,32)
4740 FOR I=1 TO 10
4750 CALL HCHAR(3,11+I,32)
4760 NEXT I
4770 IF ST=0 THEN 4650 ELSE 330
4780 REM SUB PUNKTE
4790 CALL HCHAR(24,F*10-5,32,6)
4800 IF SP(F,1)>=0 THEN 4820
4810 SP(F,1)=0
4820 P=SP(F,1)
4830 FOR I=5 TO 0 STEP -1
4840 PT=INT(P/10↑I)
4850 P=P-PT*10↑I
4860 CALL HCHAR(24,F*10-I,PT+48)
4870 NEXT I
4880 RETURN
4890 REM SUB CRASH
4900 FOR I=0 TO 30
4910 CALL SOUND(-50,110,I,114,I,118,I,-5,I)
4920 NEXT I
4930 SP(F,1)=SP(F,1)-2*(SP(F,2)+BL)
4940 GOSUB 4790
4950 RETURN
4960 REM SUB SAMMELN
4970 CALL SOUND(50,330,0)
4980 CALL SOUND(50,220,0)
4990 FOR I=0 TO 30 STEP 3
5000 CALL SOUND(-50,277,I)
5010 NEXT I
5020 SP(F,1)=SP(F,1)+79
5030 SP(F,2)=SP(F,2)+1
5040 GOSUB 4790
5050 RETURN
```

* Schweinerei *
* made in Hamburg *

SOFTWARE IM TEST



Rabbit Trail

In diesem Spiel ist man zur Abwechslung einmal ein Kaninchen und hat als solches eine Reihe von Feinden. Nach der Anleitung — eine deutsche haben wir leider nicht gefunden, also hielten wir uns an die englische — muß man darauf achten, weder von Wieseln noch von Falken getötet zu werden. Und dann sind da noch die Autos und die Fallen!

Die Wiesel treten dann auch gleich in Massen auf. Sie kommen unserem Kaninchen entgegengelaufen, die einzige Rettung: ein Sprung drüberweg.

Plötzlich wird der Spieler von einem seltsamen Rascheln aufgeschreckt. Ein Falke stürzt sich auf das arme Kaninchen. Also ab in die Höhle oder auf einen anderen Pfad, um sich vor dem mörderischen Feind zu retten.

Mühsam nähret sich nicht nur das Eichhörnchen — auch unser armes Kaninchen findet im ersten Bild nur magere zwei Karotten, die auch noch im Sprung gefressen werden müssen. Jede auf diese anstrengende Weise verspeiste Karotte beschert dem Spieler dafür aber auch 100 Punkte. Sind keine Karotten mehr übrig, folgt das nächste Bild. Hier gibt es ganze sechs Möhren zu fressen. Das ist aber auch nötig, denn das Kaninchen muß sich ordentlich stärken: unsere speziellen Feinde, die Wiesel, treten nun gelegentlich in Dreiergruppen auf. Und über mehr als eines kann unser Kaninchen nicht springen.

Hat das Kaninchen auch dieses Bild überstanden und alle Karotten gefressen, kommt noch ein drittes Bild. Die Wiesel greifen jetzt nicht mehr nur von vorn an, sondern aus allen Richtungen. Wieder gilt es, einige Möhren zu fressen, und der Weg ans Ziel ist sehr lang.

Bemerkenswert ist, daß die Wiesel nicht nur mit ihren scharfen, kleinen Zähnen unser Kaninchen totbeißen. Nein, das bedauernswerte Tier fällt auch tot um, wenn sein Sprung etwas zu kurz geraten ist und es einem Wiesel auf den Schwanz tritt.

Die Graphik ist dürftig: ein paar Pfade, ein kleines, recht niedliches Kaninchen, viele Wiesel.

Der Sound beschränkt sich auf ein 'Hüpf-Geräusch' und einen 'Todesschrei' des Kaninchens sowie ein 'Flapp-Flapp' des Vogels.

Ob es ein Ziel gibt, das man erreichen muß, haben wir nicht herausfinden können. Denn spätestens im dritten Bild hat die gewaltige Übermacht der Wiesel unser kleines Kaninchen um das letzte seiner drei Leben gebracht. Daran liegt es wohl auch, daß wir die in der Anleitung erwähnten Autos und Fallen nie zu Gesicht bekommen haben. Ein Kaninchenleben ist kurz und schwer!

Alles in allem: man kauft mit diesem Modul ein einziges IC für 125,— Mark, was in keinem Verhältnis zum Spiel steht. Jeder TI-User, der sich mit TI-Extended BASIC beschäftigt, ist in der Lage, ein solches Spiel zu schreiben. Graphik, Sound und Spiellogik sind bestimmt nicht die Kriterien, die das Spiel so teuer gemacht haben, ebensowenig die Tatsache, daß 'Rabbit Trail' ein billiger Abklatsch von 'Donkey Kong' ist.

Graphik: ☹

Sound: ☹

Spielidee: ☹

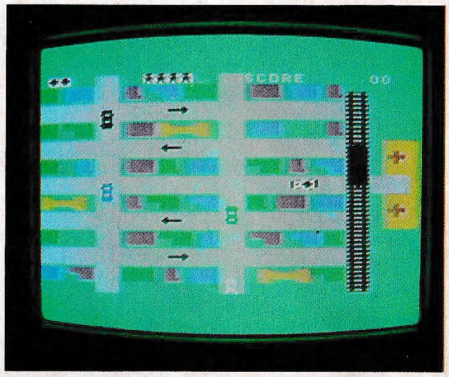
Testergebnis:

Ambulance

Zu diesem Modul haben wir eine deutsche Anleitung bekommen. Danach müssen schwerkranke Menschen in ein Krankenhaus gefahren werden. Hat man fünf Patienten nicht rechtzeitig eingeliefert oder drei Unfälle gebaut, ist das Spiel aus.

Nach Einschieben des Moduls erhielten wir die Meldung „PRESS 2 FOR AMBULANCE". Wir taten es — und wurden von einer allzu simplen Melodie begrüßt, die in einer Endlos-Schleife ständig wiederholt wird. Eiligst begannen wir das Spiel.

Im ersten Bild erscheint eine Straßenkarte, Autos fahren geschäftig durch die Straßen und ein Haus blinkt. Das bedeutet, hier ist ein Kranker abzuholen und ins Krankenhaus einzuliefern. Dabei muß man auf die anderen Autos achten, obendrein auf die Eisenbahn, die ab und zu vor dem



Krankenhaus vorbeifährt! Denn all diese Verkehrsteilnehmer nehmen keine Rücksicht auf den Rettungswagen.
Hat man dann eine Reihe von Patienten glücklich ins Krankenhaus geschafft, folgt das zweite Bild. Da gibt es nun verwinkelte Straßen und außerordentlich feindselige Autos, die nicht mehr einen leicht voraussehbaren Weg fahren. Wieder blinkt ein Haus nach dem anderen, wo Patienten abgeholt werden müssen. Ja, und das ist auch schon das ganze Spiel, hat man nämlich genug Patienten gerettet, folgt wieder das erste Bild, es beginnt von vorn.
Die Graphik: ein paar Kästchen, die Häuser darstellen sollen. Dazu ein paar Streifen für die Straßen, skizzierte Autos und ein Zug (in Draufsicht). Der Rettungswagen ist lediglich durch ein Kreuz gekennzeichnet.
Der Sound beschränkt sich, außer der nervtötenden Anfangsmelodie, auf eine Art 'Sirene', wenn ein Patient an Bord ist (die allerdings von den anderen Autos nicht beachtet wird), einen fallenden Ton, wenn ein Patient stirbt, sowie ein 'Krachen' bei einem Unfall.
Der Preis, den man für dieses allzu magere Spiel zu zahlen hat, ist mit 125,— Mark wohl als Witz aufzufassen. Jeder TI-User, der Erfahrung mit TI-Extended BASIC hat, kann sich ein solches Programm selbst erstellen. Und das macht bestimmt mehr Spaß als dieses Spiel-Modul!

Graphik: ☹

Sound: ☹

Spielidee: ☹

Testergebnis:

☹

Mr. Becker goes to Texas

Data Becker, bislang ausschließlich Commodore zugetan, was die bisher erschienenen Bücher angeht, hat mit diesem Band erstmals zarte Bande zum Texas Instruments geknüpft. Allzu billig ist die dargebotene Liebe allerdings nicht, für 49,— DM solltet ihr schon einiges erwarten dürfen.
TI-BASIC, EXTENDED-BASIC und MICROSOFT-BASIC werden verglichen, und zum Umschreiben von MICROSOFT- auf TI-BASIC findet sich hier mancher gute Tip. Unter den 14 tippfertigen Programmen reicht das Themenspektrum von Spielen wie 'Risiko', 'Schiffe versenken' oder 'Wortspielerei' bis zu Nützlicherem (Wir bekennen uns ausdrücklich dazu, auch Spielen als nützlich anzusehen!) wie Textverarbeitung, Kfz-Kostenrechnung oder Mathe-Trainer. Die Listings sind ausnahmslos sehr gut kommentiert.
Das Kapitel 'Datenverwaltung' behandelt ausführlich den Cassettenrecorder, läßt die Diskettenverarbeitung aber zu kurz kommen. Unter 'Tips und Tricks' gibt es einige gute Anregungen, Anderes ist jedoch zum Teil überholt (Passwort z.B.). Zu den Themen Graphik und Tonausgabe wird kaum mehr geboten, als im TI-Handbuch steht, zu EXTENDED-BASIC fiel dem Autor noch weniger ein.
Gut, auch für Fortgeschrittene zu empfehlen, ist Kapitel 9: Ein Programm entsteht. Die Programmerstellung mit Hilfe von Flußdiagrammen wird mit Beispielen erläutert. Das letzte Kapitel befaßt sich mit den Sprites, ist gut geschrieben und beinhaltet einen 'Sprite-Editor' als Listing.
Der Anhang enthält Tabellen, die sich ebenso im TI-Handbuch finden, eine Liste der Fehlermeldungen, aber nur für EXTENDED-BASIC, in Englisch und ohne ein Wort der Erläuterung, also wertlos, sowie je 5 Seiten eines Bildschirm- und Sprite-Entwurfsblattes (im Zeitalter des Fotokopierens hätte man den Platz besser für ein Register verplant, das wir nun wirklich für wichtig halten).
Anmerken möchten wir noch, daß das Buch vor Neuauflage noch einmal in Hinsicht auf Rechtschreibung kontrolliert werden sollte.
Die Pluspunkte, die zweifellos an 'TI-99 Tips und Tricks' zu entdecken waren, rechtfertigen unserer Meinung nach nicht die Investition von 49.— DM.

TI-99 Tips und Tricks

Schwinn
Verlag DATA BECKER
49.- DM
303 Seiten
ISBN 3-89011-006-1

TI'S TELEFONBUCH

Wer einen Computer hat, braucht kein Adreßbuch mehr. Daten aller Art: leicht zugänglich durch unser beinahe professionelles Programm.

Das Problem, mit dem TI 99/4A und einem Cassetten-Recorder Daten zu verwalten, ist nicht einfach zu lösen. Die meisten Programme zu diesem Thema sind entweder so umfangreich, daß die 16 Kbyte RAM kaum noch Daten aufnehmen können. Oder sie sind so langsam, daß man es am besten gleich wieder löscht und sich seine Daten auf einem Zettel aufschreibt. Oder sie haben die Daten fest in DATA-Zeilen im Programm: Die Daten sind dann während des Programmlaufs nicht veränderbar.

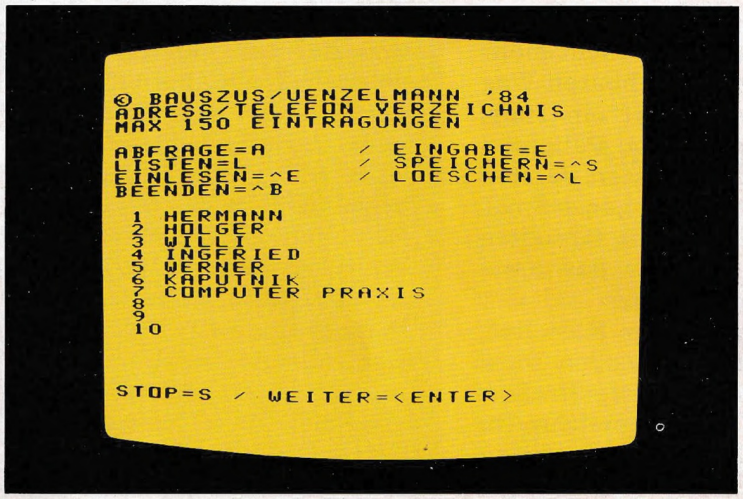
Uns liegt hier ein Listing vor, das unsere Forderungen (schnell, variabel, große Eintragungszahl) noch gerade so eben erfüllt. Die im TI-XBASIC verfügbaren 13928 Bytes werden in diesem Programm folgendermaßen belegt:

2934 Bytes Programm
10994 Bytes frei für Daten.

Das reicht für 150 Eintragungen zu je 3x15 Zeichen (Name, Adresse, Telefon).

Der erste Kniff bei der Sache ist, daß die Daten nur dann auf Cassette geschrieben werden, wenn das Programm unterbrochen oder das Kommando 'Speichern' (^ S) gegeben wird.

Wird das Programm irgendwann neu gestartet, müssen die Daten, die ausgeladen wurden, mit einem ^ E— (Einlesen) Kommando neu eingeladen werden. Da jeweils 3 Eintragungen in einem Satz à 192 Zeichen auf Cassette gespeichert und nicht mehr Sätze als nötig geschrieben werden, bleibt der Zeitaufwand vertretbar. Trotzdem ist und bleibt die Datenverwaltung mit Cassetten ein Geduldspiel.



Oben das Hauptmenü, in der Mitte die Ausgabe und unten ein Zwischen-Menü. Der Bildschirm ordentlich sortiert.

Der zweite Kniff ist, daß die Daten in einem dimensionierten Feld (= Array) gespeichert werden. Ein Array könnt ihr euch als eine Kette von durchnumerierten Schachteln vorstellen. In der Schachtel 1 steht die Eintragung 1, in Schachtel 2 die Eintragung 2 usw. Das letzte Element der Kette wird mit dem DIM-Kommando festgelegt.

In unserem Listing haben wir ein zweidimensionales Feld. Ihr müßt euch das wie die Felder eines Koordinatensystems vorstellen. Die Zeilen sind die Eintragungs-Nummern, die Spalten die eigentlichen Eintragungen.

Arrays sind eine schöne Sache, sind sie doch sehr übersichtlich und die Daten schnell abrufbar. Jede Eintragung wird mit einer Zahl versehen. Man kann jetzt auf die Daten entweder mit dieser Zahl gezielt zugreifen oder mit einer Schleife in dem Array nach einer bestimmten Eintragung suchen. Mit den kleinen Rechnern, wie wir sie benutzen, und mit der Sprache BASIC gibt es wohl keine schnellere Möglichkeit.

Natürlich können in einem Array auch andere Daten als Namen, Adressen und Telefon-Nummern gespeichert werden. Wir sind aber der Meinung, das dies ein gelungenes Beispiel ist.

Zum Listing

Zeile 100

Dimensionierung des Arrays, Abbruchschutz

110 - 130

Menü des Programms sowie Abfrage der Eingaben über CALL KEY

140 - 210

Neue Eintragung ins Array
140 Eingabe des Namens. Wird ein Leerstring gegeben, Rücksprung in Menü. Der Name kann maximal 15 Zeichen umfassen.
150-160FOR NEXT Schleife, die die erste Spalte durchsucht. Begrenzt ist die Schleife durch die Zahl der letzten Eintragung (ANZ). Ist vor dieser neuen Ein-

tragung eine alte mit einer Nummer kleiner ANZ gelöscht worden, so wird die neue an dieser Stelle eingefügt. Sonst wird geprüft, ob die letzte Eintragung kleiner 150 ist (LAEN). Ist dies der Fall, wird die neue Eintragung am Ende angehängt.

170LOE=0 heißt: Daten-Array wurde verändert.

180-210 Adresse und Telefonnummer werden eingegeben. Wird hier jeweils ein Leerstring gegeben, wird in das Array ein Stern für die Adresse und die Telefon-Nummer gesetzt. Anschließend erfolgt der Rücksprung ins Menü.

220 - 300

Abfrage der Eintragung

220 Der gesuchte Name wird eingegeben. Wird ein Leerstring gegeben, erfolgt ein Rücksprung ins Menü. Statt des kompletten Namens kann auch der erste Buchstabe, gefolgt von einem Stern, eingegeben werden. Dies wird eine Wild-Card (Joker) genannt. In diesem Fall wird das Programm in Zeile 270 fortgesetzt.

230-260FOR NEXT Schleife, die die erste Spalte des Arrays nach dem angegebenen Namen durchsucht. Wird er nicht gefunden, erfolgt eine entsprechende Meldung und ein Rücksprung ins Menü. Sonst werden der Name, die Adresse und die Telefonnummer der Eintragung ausgegeben und gefragt, ob es die gewünschte ist (OK?). Wird 'J' für 'Ja' geantwortet, springt das Programm zurück ins Menü. Wird 'N' für 'Nein' gegeben, sucht das Programm, ob eine weitere Eintragung mit gleichem Namen vorhanden ist.

270-330 Hier wird die Suche nach Eingabe einer Wild Card (siehe Zeile 220) durchgeführt. Das Prinzip ist dasselbe wie in den Zeilen 230-260. Allerdings werden hier alle Namen ausgegeben, die mit dem angegebenen Buchstaben beginnen. Die folgenden Buchstaben werden durch den Stern, d.h. die Wild Card, symbolisiert. Dieser Stern kann für jede mögliche Buchstabengruppe stehen.

310 - 370

Liste der eingetragenen Namen

310 Der Anfang der gewünschten Liste wird gesetzt. Wird eine Eingabe kleiner 1 gemacht, beginnt die Liste mit dem 1. Element des Arrays. Bei einer Eingabe größer ANZ beginnt sie mit ANZ.

320 Das Ende der gewünschten Liste wird gesetzt. Wird eine Eingabe kleiner Liste-Anfang gemacht, endet die Liste mit dem zehnten Element nach Liste-Anfang. Bei einer Eingabe größer ANZ endet die Liste mit ANZ.

330-370 Die in dem Array eingetragenen Namen werden in Zehner-Blöcken ausgegeben. Ist das Ende der gewünschten Liste nicht erreicht, oder ist der gesuchte Name nicht gefunden worden, kann die Ausgabe der Liste mit der ENTER-Taste fortgesetzt werden. Der nächste Zehner-Block wird ausgegeben. Mit 'S' wie 'Stop' wird die Ausgabe abgebrochen.

380 - 420

Ausladen auf Cassette / Abbruch des Programms

380 Wird im Menü ^B (Beenden) gedrückt, prüft das Programm, Ob LOE=1 ist. Wenn ja, erfolgt ein Sprung in Zeile 540. Sind die Eintragungen im Array geändert worden (neue hinzugekommen/ alte gelöscht), muß der Inhalt des Arrays erneut auf Cassette abgespeichert werden. Dies geschieht, wenn im Menü entweder ^B oder — ^S (Speichern) gedrückt wird. Im ersten Fall (^B) wird das Programm anschließend in den Zeilen 540-560 unterbrochen. Die Eintragungen des Arrays werden jetzt in Sätzen zu je 192 Zeichen ausgeladen. Im ersten Satz steht die Zahl der letzten Eintragung (ANZ).

390-420 Die FOR NEXT Schleife lädt jeweils 3 Eintragungen (jede besteht aus Name/Adresse/Telefonnummer) in einem Satz aus. Zur Kontrolle werden die ausgeladenen Daten auch auf dem Bildschirm ausgegeben. In Zeile 420 wird die Datei geschlossen und dann LOE wieder gleich 1 gesetzt, damit das Programm erkennt, daß die neueste Version des Arrays auf Cassette gespeichert ist. Außerdem wird noch einmal geprüft, ob das Programm beendet werden soll. Wenn ja, erfolgt ein Sprung in Zeile 540, sonst ein Rücksprung ins Menü.

430 - 470

Einladen der Daten von Cassette

Eine Datei zum Einladen der auf Cassette gespeicherten Daten wird eröffnet. Der erste Satz, der die Zahlen der Eintragung enthält, wird eingelesen. Dann werden in einer FOR NEXT Schleife die Daten eingelesen, die im Array untergebracht werden sollen. Die Schleife ist durch die zuerst eingelesene Zahl (ANZ) begrenzt. Gleichzeitig werden die Daten auf dem Bildschirm ausgegeben. Zeile 470 schließt die Datei, dann erfolgt ein Rücksprung ins Menü.

480 - 530

Löschen einer Eintragung

480 Der Name der zu löschenden Eintragung wird eingegeben. Er muß nicht voll ausgeschrieben sein. Eine Wild Card ist hier nicht erlaubt. Wird ein Leerstring eingegeben, springt das Programm ins Menü zurück.

490-530 Die FOR NEXT Schleife durchsucht das Array nach dem angegebenen Namen. Wird er nicht gefunden, erfolgt eine entsprechende Meldung und der Rücksprung ins Menü. Sonst werden die entsprechenden Eintragungen aus dem Array ausgegeben und eine Bestätigung verlangt, daß es die richtigen sind. Wird mit 'N' für 'Nein' geantwortet, sucht das Programm nach einer weiteren Eintragung mit dem gleichen Namen.

Wird 'J' für 'Ja' geantwortet, werden der Name, die Adresse und die

Telefonnummer aus dem Array gelöscht. Dann setzt das Programm LOE gleich 0 und springt zurück ins Menü.

540 - 560

Unterbrechung des Programms Wenn B gedrückt wurde, wird das Programm in der Zeile 550 durch ein BREAK unterbrochen. Das Array wird durch das BREAK-Kommando nicht gelöscht. Die Daten bleiben erhalten, solange der Rechner nicht abgeschaltet wird oder neue Zeilen editiert werden. Mit CON oder CONTINUE kann das Programm fortgesetzt werden.

Noch eine Bemerkung zum Menü: Alle Buchstaben müssen als Großbuchstaben eingegeben werden. ^B bedeutet, daß erst die CONTROL (CTRL)-Taste gedrückt und gehalten werden muß, dann wird zusätzlich der entsprechende Buchstabe (hier B) gedrückt.

TI-TELEFONBUCH

```
100 ON WARNING NEXT :: ON BREAK NEXT :: LOE,ANZ=1 ::
    LAEN=150 :: DIM T$(150,3)
110 CALL CHAR(96,"3C4299A1A199423C")::
    DISPLAY AT(1,1)ERASE ALL:
    "{CHR$96} BAUSZUS/UENZELMANN ^84":
    "ADRESS/TELEFON VERZEICHNIS":
    "MAX 150 EINTRAGUNGEN"
120 DISPLAY AT(5,1):"ABFRAGE=A","/ EINGABE=E",
    "LISTEN=L","/ SPEICHERN={CHR$94}S",
    "EINLESEN=[CHR$94}E","/ LOESCHEN={CHR$94}L",
    "BEENDEN={CHR$94}B"
130 CALL KEY(5,K,S)::
    IF S=0 THEN 130 ELSE DISPLAY AT(22,1): : : ::
    IF K=65 THEN 220 ELSE IF K=76 THEN 310 ELSE
    IF K=147 THEN 380 ELSE IF K=133 THEN 430 ELSE
    IF K=140 THEN 480 ELSE IF K=130 THEN 380 ELSE
    IF K<>69 THEN 130
140 DISPLAY AT(24,1):"NAME:" ::
    ACCEPT AT(24,7)SIZE(15):N$ ::
    IF N$="" THEN DISPLAY AT(24,1):"" :: GOTO 120
150 FOR I=1 TO ANZ :: IF T$(I,1)="" THEN 170
160 NEXT I ::
    IF ANZ=LAEN THEN DISPLAY AT(23,1):"TABELLE IST
    VOLL.{11 SPACE}KEINE EINGABE MEHR MOEGLICH." ::
    GOTO 120 ELSE I,ANZ=ANZ+1
170 LOE=0 :: T$(I,1)=N$
180 DISPLAY AT(24,1):"ADRESSE:" ::
    ACCEPT AT(24,10)SIZE(15)BEEP:A$ ::
    IF A$="" THEN A$="*"
190 T$(I,2)=A$
200 DISPLAY AT(24,1):"TELEFON:" ::
    ACCEPT AT(24,10)SIZE(15)BEEP:TL$ ::
    IF TL$="" THEN TL$="*"
210 T$(I,3)=TL$ :: DISPLAY AT(24,1):"" :: GOTO 120
220 DISPLAY AT(24,1):"NAME: " ::
    ACCEPT AT(24,7)SIZE(15)BEEP:N$ ::
    IF N$="" THEN DISPLAY AT(24,1):"" :: GOTO 120 ::
    ELSE IF SEG$(N$,2,1)="*" THEN 270
230 FOR I=1 TO ANZ :: IF T$(I,1)<>N$ THEN 260
240 FOR A=1 TO 3 :: DISPLAY AT(21+A,1):T$(I,A)::
    NEXT A
250 DISPLAY AT(24,24):"OK?" ::
    ACCEPT AT(24,28)VALIDATE("JN")SIZE(1):X$ ::
    IF X$="J" THEN 120 ELSE DISPLAY AT(22,1): : :
260 NEXT I :: DISPLAY AT(24,1):"NICHT GEFUNDEN" ::
    GOTO 120
270 FOR I=1 TO ANZ ::
    IF SEG$(T$(I,1),1,1)<>SEG$(N$,1,1)THEN 300
280 FOR A=1 TO 3 :: DISPLAY AT(21+A,1):T$(I,A)::
    NEXT A
290 DISPLAY AT(24,24):"OK?" ::
    ACCEPT AT(24,28)VALIDATE("JN")SIZE(1):X$ ::
    IF X$="J" THEN 120 ELSE DISPLAY AT(22,1): : :
300 NEXT I :: DISPLAY AT(24,1):"NICHT GEFUNDEN" ::
    GOTO 120
310 DISPLAY AT(24,1):"LISTE START:" ::
    ACCEPT AT(24,14)SIZE(3)VALIDATE(DIGIT)BEEP:ST ::
    IF ST<1 THEN ST=1 ELSE IF ST>ANZ THEN ST=ANZ
320 DISPLAY AT(24,1):"LISTE STOP:" ::
    ACCEPT AT(24,13)SIZE(3)VALIDATE(DIGIT)BEEP:STO ::
    IF STO<ST THEN STO=ST ELSE IF STO>ANZ THEN STO=ANZ
330 FOR I=ST TO STO STEP 10
340 FOR Z=0 TO 9 ::
    DISPLAY AT(Z+10,1):I+Z;T$(I+Z,1):: NEXT Z
350 DISPLAY AT(24,1)BEEP:"STOP=S / WEITER=<ENTER>"
360 CALL KEY(0,K,S)::
    IF S=0 THEN 360 ELSE IF K=83 THEN 110 ELSE
    IF K<>13 THEN 360
370 NEXT I :: GOTO 330
380 IF K=130 AND LOE=1 THEN 540 ELSE CALL CLEAR ::
    OPEN #1:"CS1",INTERNAL,OUTPUT,FIXED 192 ::
    PRINT #1:ANZ
390 FOR I=1 TO ANZ STEP 3
400 PRINT #1:T$(I,1),T$(I,2),T$(I,3),T$(I+1,1),
    T$(I+1,2),T$(I+1,3),T$(I+2,1),T$(I+2,2),T$(I+2,3)
410 PRINT I:T$(I,1):T$(I,2):T$(I,3):
    :I+1:T$(I+1,1):T$(I+1,2):T$(I+1,3):
    :I+2:T$(I+2,1):T$(I+2,2):T$(I+2,3): :
420 NEXT I :: LOE=1 :: CLOSE #1 ::
    IF K=130 THEN 540 ELSE 110
430 CALL CLEAR ::
    OPEN #1:"CS1",INTERNAL,INPUT,FIXED 192 ::
    INPUT #1:ANZ
440 FOR I=1 TO ANZ STEP 3
450 INPUT #1:T$(I,1),T$(I,2),T$(I,3),T$(I+1,1),
    T$(I+1,2),T$(I+1,3),T$(I+2,1),T$(I+2,2),T$(I+2,3)
460 PRINT I:T$(I,1):T$(I,2):T$(I,3):
    :I+1:T$(I+1,1):T$(I+1,2):T$(I+1,3):
    :I+2:T$(I+2,1):T$(I+2,2):T$(I+2,3): :
470 NEXT I :: CLOSE #1 :: GOTO 110
480 DISPLAY AT(23,1):"WELCHER NAME SOLL GELOESCHT
    WERDEN?" :: ACCEPT AT(24,9)SIZE(15):N$ ::
    IF N$="" THEN DISPLAY AT(23,1):"":"" :: GOTO 120
490 FOR I=1 TO ANZ :: IF T$(I,1)<>N$ THEN 520
500 FOR A=1 TO 3 :: DISPLAY AT(21+A,1):T$(I,A)::
    NEXT A :: DISPLAY AT(24,24):"OK?"
510 ACCEPT AT(24,28)SIZE(1)VALIDATE("JN")BEEP:X$ ::
    IF X$="J" THEN 530
520 NEXT I :: DISPLAY AT(22,1):"EINTRAGUNG
    NICHT GEFUNDEN--{12 SPACE}NICHT GELOESCHT": ::
    GOTO 120
530 FOR A=1 TO 3 :: T$(I,A)="" ::
    DISPLAY AT(21+A,1):"" :: NEXT A :: LOE=0 ::
    GOTO 120
540 CALL CLEAR
550 BREAK
560 GOTO 110 ■
```

Woher nehmen und nicht stehlen?

Neue Peripherie ist noch immer viel zu wenig in Sicht. Machen wir uns darum einen normalen Cassetten-Recorder für unseren TI einsetzbar!

Wer sich einen Computer zulegt, wird früher oder später den Wunsch verspüren, Software nicht nur für die Zeit parat zu haben, in der der Rechner eingeschaltet ist. Oder der neue User hat, durch hilfreiche Unterstützung der Werbung, festgestellt, daß es Software fertig zu kaufen gibt. Und zwar entweder auf Disketten oder auf Cassetten.

Damit sind wir beim Thema. Da Texas Instruments so freundlich war, seine Lager zu räumen, bevor sich die Firma aus der Home-Computer-Branche zurückzog, sitzen heute viele User ohne Massen-Speicher an ihrem Rechner. Disketten-Laufwerke und die dazugehörigen Controller sind für die Expansion-Box nicht mehr zu bekommen — wenn man überhaupt noch das Glück hatte, eine Box zu ergattern.

Entsprechender Ersatz von anderen Herstellern, der mit unseren TI 99/4A kompatibel ist, ist auf dem Markt nicht erhältlich. Also greife man auf den Cassetten-Recorder zurück.

Sofort ergibt sich das nächste Problem: Anschluß-Kabel Rechner-Recorder sind nicht mehr oder nur zu horrenden Preisen zu bekommen. Ein Grund für uns, euch eine Schaltskizze anzubieten. Anhand der hier abgedruckten Skizze könnt ihr euch dieses Kabel selber bauen. Alles was ihr braucht, bekommt ihr im Fachhandel:

▷3 Klinken-Stecker 3,5 mm, 2-polig
▷2 Klinken-Stecker 2,5 mm, 2-polig
▷1 V9-Stecker (female)
▷1,5 m 6-phasiges Kabel
und den obligatorischen Lötkolben.

Das Kabel reicht, um die Möglichkeiten des TI 99/4A voll zu nutzen, er ist nämlich in der Lage mit zwei Recordern zu arbeiten! Die Pins des V9-Steckers sind wie folgt belegt:

Pin 1 und 2: Fernsteuerung Recorder 1 (CS1)
Pin 3 und 5: Aufnahme Recorder 1 und Recorder 2 (CS1 und CS2)
Pin 6 und 7: Fernsteuerung Recorder 2 (CS2)
Pin 8 und 9: Abspielen Recorder 1 (CS1)

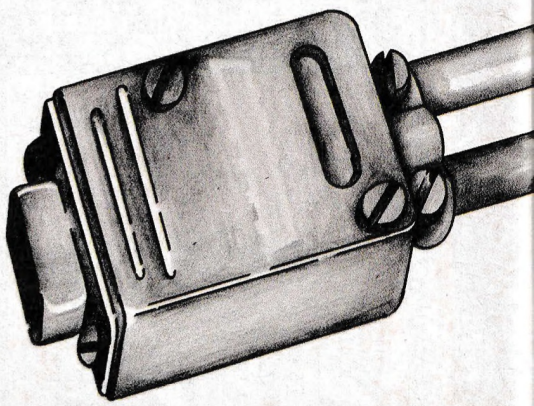
Pin 4 ist nicht belegt. Wenn ihr nur einen Recorder anschließen wollt, braucht ihr die Pins 6 und 7 auch nicht zu belegen. Ihr kommt dann mit einer einzigen Leitung aus.

Beachten müßt ihr noch, daß die Pins 3 und 9 jeweils die Abschirmung für Aufnahme bzw. Abspielen sind. Wie die Fernsteuerung gepolt werden muß, hängt vom anzuschließenden Recorder ab. Hier hilft nur das Experiment.

Welcher Recorder?

Eine nicht selten gehörte Frage ist, welcher Recorder denn nun der beste ist. Unsere Erfahrung: Ein einfacher Mono-Recorder reicht völlig. Solche Geräte sind ab ca. 60,— DM im Handel. Alle Versuche mit teuren Stereo-Geräten oder sogar Spulen-Tonbandgeräten verliefen negativ.

Der zweite Punkt, der beachtet werden muß: nur gutes Bandmaterial benutzen. Billig-Cassetten oder alte Musik-Cassetten haben nur ein Ergebnis, Frust. Diese alten Bänder verlieren nämlich mit der Zeit ihre Magnet-Schicht und damit die darin enthaltene Information. Bei Musik-Aufnahmen fällt das unter Umständen noch nicht einmal auf. Aber der Computer mault sofort "ERROR IN DATA DETECTED", Fehler in den Daten gefunden. Das bedeutet, die gespeicherten Daten sind zum Teufel. Was dem User bleibt, ist mal wieder Frust.

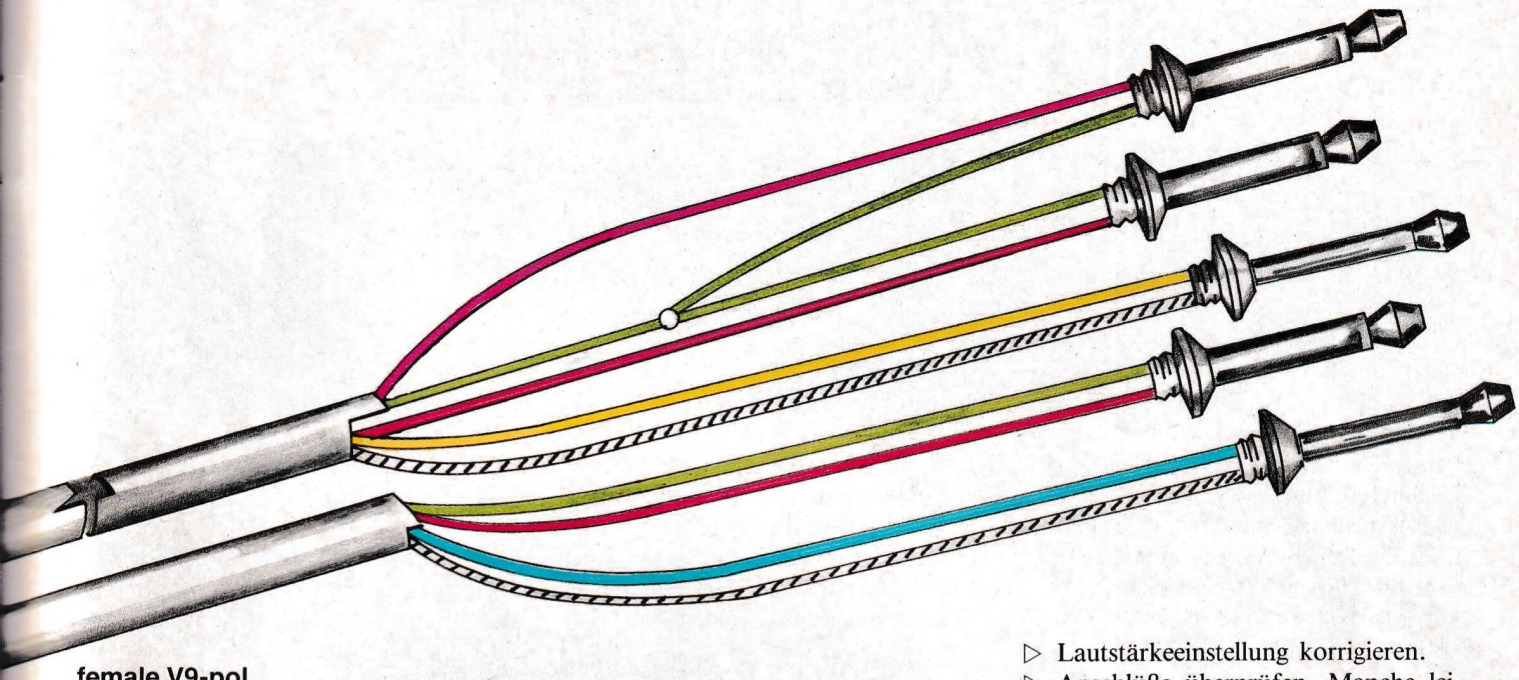


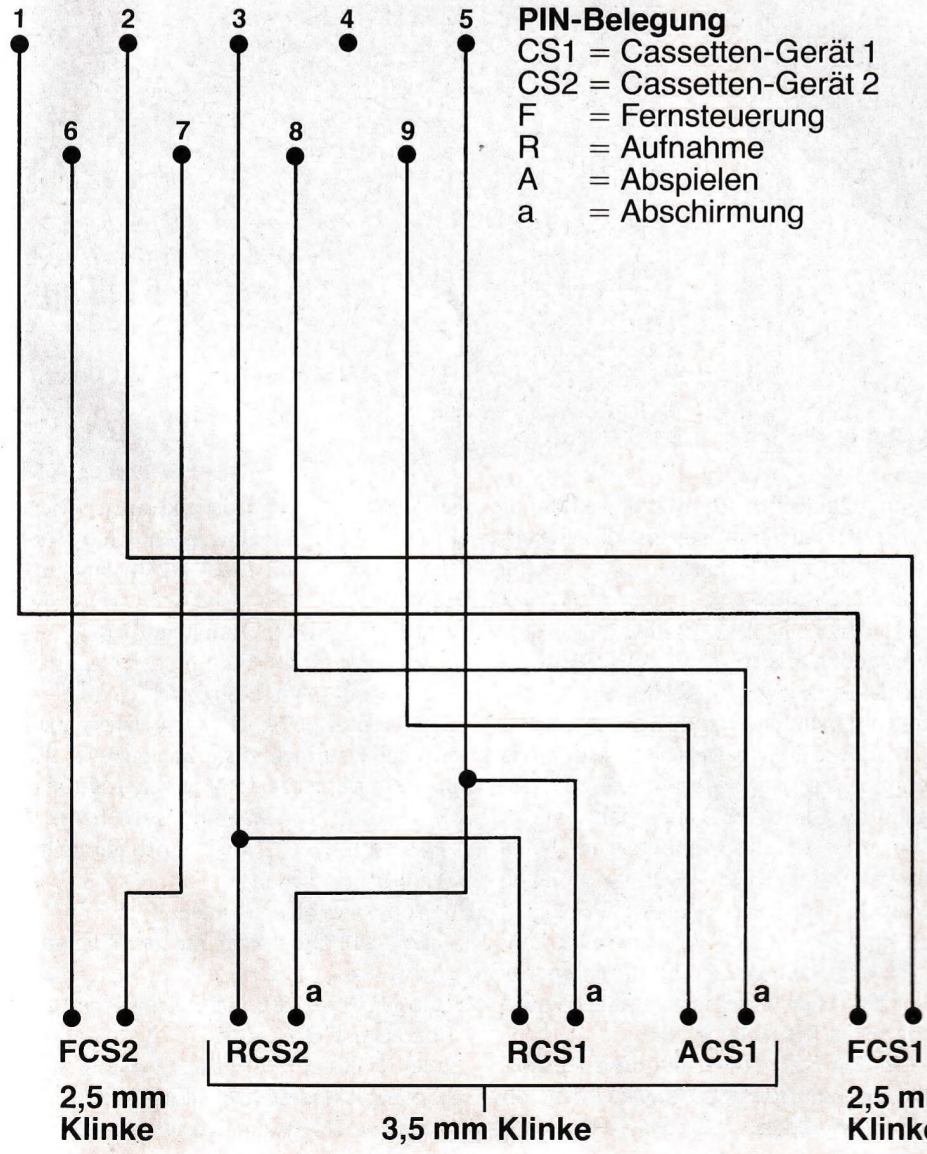
So sieht unser Anschlußkabel in der Version für zwei Cassettenrecorder aus.

Aus diesem Grund sollte man auch von allen wichtigen Programmen eine "BackUp"-Kopie, das heißt eine Sicherungs-Kopie, anfertigen. Und zwar auf einer anderen und nicht auf der gleichen Cassette. Wenn euch jetzt ein Programm durch Altersschwäche der Cassette ins Nirwana abhaut, nehmt ihr locker eure BackUp aus dem Regal und könnt weiterarbeiten (oder spielen).

Cassetten sollte man übrigens nicht auf Magneten oder in der prallen Sonne lagern. Hitze und Magnetfelder haben die unangenehme Eigenheit, Tonband-Material zu löschen und Wärme verformt außerdem das Gehäuse einer Cassette unter Umständen bis hin zur Unbrauchbarkeit.

Und dann gibt es Tage, da will sich ein Programm, das man schon so oft erfolgreich geladen hat, nicht in den Rechner holen lassen. Auch die BackUp hilft nicht. Auch hierfür haben wir einige Tips für euch. Helfen auch die nicht, weiß keiner mehr weiter.





▷ Lautstärkeeinstellung korrigieren.
▷ Anschlüße überprüfen. Manche leiern aus.
▷ steht der Recorder zu dicht am Fernseher (Monitor)? Die Ausstrahlungen der Bildröhre können den Ladevorgang beeinträchtigen.
▷ Tonköpfe reinigen. Dabei sollte man lieber Isopropanol, ein Alkohol, den ihr in der Apotheke bekommt und Watte-Stäbchen benutzen als teure Reinigungs-Bänder, die euch unter Umständen den Tonkopf zusätzlich mit Schmutzpartikeln verkleben.
▷ Als letzte Maßnahme: Vorsichtige Korrektur der Spurlage

Im Normalfall hat der TI 99/4A-User aber keine Probleme, Programme auf Cassetten zu speichern und sie von dort wieder einzuladen. Wenn ihr darauf achtet, daß euer Bandmaterial in Ordnung ist, werdet ihr so manches interessante Daten-Verwaltungs-Programm schreiben können. Denn mit dem TI kann man nicht nur Programme sondern auch Daten auf Cassette abspeichern. Natürlich gehört auch etwas Geduld dazu, denn die Baud-Rate, das ist die Zahl der übertragenen Bits pro Sekunde, ist bei dem TI 99 leider ziemlich niedrig.

Wer aber diese Geduld nicht aufbringen kann oder will, der muß auf andere Weise Geduld beweisen. Denn die angekündigten Floppy-Controller sind leider immer noch nicht auf dem Markt. Und dabei wird es wohl auch noch eine ganze Weile bleiben. ■

H.U.

EIMERKETTE

Der Zauberlehrling in XBASIC

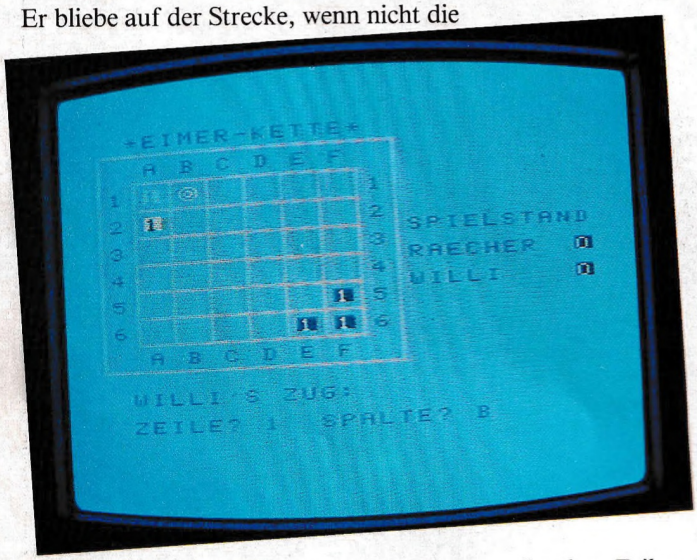
Bei diesem Listing handelt es sich um eine Art Brettspiel. Jedes Feld des Brettes bedeutet einen Eimer, und es ist Ziel des Spiels, die Eimer unter Kontrolle zu bekommen. Es gilt die einfache Regel: "Ein Eimer ist mein, wenn was von mir drinliegt". Man braucht also nur Zug für Zug an die Kübel zu spenden, und der TI zeigt brav an, wieviel im Eimer liegt und welchem der beiden Spieler es gehört.

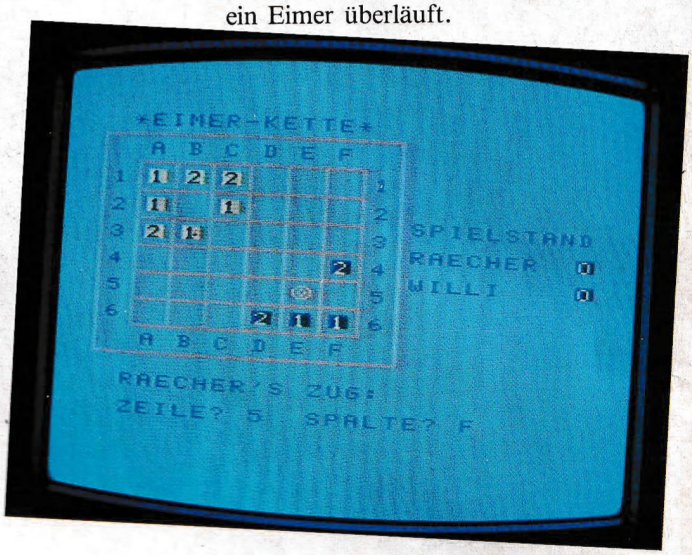
Prima, nur — wo bleibt da der Witz? Er bliebe auf der Strecke, wenn nicht die Eimer die magische Eigenschaft hätten, überzulaufen, sobald sie den Rand voll haben. Man kann also seinen Obolus nach und nach an freie Eimer entrichten, oder man kann einen, den man schon in seinen Besitz gebracht hat, durch steten Tropfen zum Überquellen bringen. Und das geht rasch. In der Ecke stehende Behälter gehen schon hoch, wenn man sie ein zweites Mal bedenkt, Eimer am Rand werden bei der dritten Füllung zu Springbrunnen und die Mittelfeldkübel reagieren auf die Anzahl 4 allergisch. Dann nämlich speien sie ihren Inhalt in die Nachbareimer, und bringen diese samt gegnerischem Inhalt in den eigenen Besitz. Mit etwas Geschick erreicht man scheinbar endlose Ketten-Reaktionen.

Das Spiel kann zu zweit oder allein gegen den TI 99 gespielt werden, oder ihr laßt den Rechner einfach gegen sich selbst antreten.

Zum Listing

Ihr braucht einen TI 99/4A und Extended BASIC. Das Spiel ist mit 8181 Bytes relativ kurz, wenn man bedenkt, was es leistet. Achtet bitte darauf, daß einige Zeilen sowohl Groß- als auch Klein-Buchstaben enthalten. Sie müssen genau so eingegeben werden!





In den Zeilen 100—200 werden die Arrays (Felder) initialisiert, das Bezugs-Array belegt, die Farben und die Character gesetzt.

In den Zeilen 210—350 wird das Spiel vorbereitet. Es werden, wenn nötig, Instruktionen gegeben, die Spieler-Anzahl und ihr Namen erfragt.

In Zeile 360 beginnt das Hauptprogramm. Es ist zwei-geteilt. Der erste Teil geht bis Zeile 580. Hier habt ihr das 'normale Spiel' Mann gegen Mann (oder Frau gegen Frau oder so...). Wenn ihr gegen den TI spielt oder den Rechner gegen sich selbst antreten laßt, verzweigt sich das Programm in die Zeile 1290. Hier findet ihr die 'Automatik'. Sie reicht bis Zeile 1570.

In den Zeile 590—660, die auch noch zum Hauptprogramm zählen, wird das Ende des Spieles mit der Feststellung, wer gewonnen hat, abgehandelt. Das Programm bleibt hier stehen, bis eine Taste gedrückt wird. Das Spiel wird mit einem STOP abgebrochen, wenn ihr 'N' drückt. Sonst kommt sogleich die Revanche-Runde.

Es folgen die Sub-Routinen

Zeile 670— 780: Das Spielbrett wird aufgebaut.

Zeile 790— 850: Regelt die Wasserzuteilung an die Eimer.

Zeile 860—1030: Hier wird geprüft, ob ein Eimer überläuft.

Zeile 1040—1240: Instruktionen. Diese Sub-Routine hat eine eigene Sub-Routine in den Zeilen 1250—1270, um 'umzublättern'.

Zeile 1580—1600: Punkteaddierer.

Zeile 1610—1640: Füllung der Nachbar-Eimer bei einem Überlauf.

Zeile 1650—1670: Hier sind zwei 'akustische' Routinen zusammengefaßt. Die erste, in der Zeile 1660, erzeugt die kleine Melodie, die zweite, in Zeile 1670, ist eine einfache aber oftgebrauchte Tonfolge.

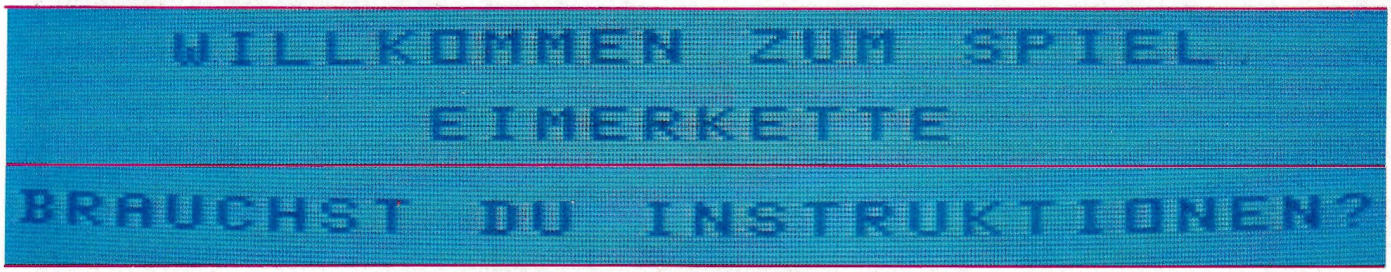
Zum Schluß die Daten

Zeile 1690: Die Daten für das Referenz-Array.

Zeile 1700—1720: Die Strings für die Character-Definierung.

Zeile 1730—1790: Die Frequenz und Tonlängen für die Musik.

Und nun: Hoch die Tassen! Äh, Eimer.




```
100 REM INITIALISIERUNG
110 CALL CLEAR :: CALL SCREEN(5)::
    DISPLAY AT(12,9):"EIMERKETTE" :: RANDOMIZE
120 GOSUB 1660
130 DIM W(6,6),B(6,6),O(6,6),P(6,6),CN$(2),V(2),D(2)
140 FOR N=1 TO 6 :: READ W(N,1):: W(N,6)=W(N,1)::
    FOR J=2 TO 5 :: W(N,J)=W(N,1)+1
150 NEXT J :: NEXT N :: CALL COLOR(14,10,1)
160 DEF RN(X)=INT(RND*X+1)
170 CALL COLOR(3,12,2):: CALL COLOR(9,14,1)::
    CALL COLOR(10,14,1):: CALL COLOR(13,2,12)
180 FOR BU=128 TO 135 :: CALL CHARPAT(BU-80,A$)::
    CALL CHAR(BU,A$):: CALL CHAR(BU-16,A$):: NEXT BU
190 FOR N=97 TO 107 :: READ N$ :: CALL CHAR(N,N$)::
    NEXT N
200 CALL CHAR(136,"0000001818")::
    CALL CHAR(137,"00003C425A5A423C")::
    CALL CHAR(138,"3C4299A5A599423C")
210 REM SPIELVORBEREITUNG
220 DISPLAY AT(10,4):"WILLKOMMEN ZUM SPIEL"
230 FOR BU=1 TO 3 :: CALL SOUND(200,220,4)::
    CALL SOUND(400,440,2):: NEXT BU
240 DISPLAY AT(22,1):"BRAUCHST DU INSTRUKTIONEN?" ::
    CALL KEY(3,KEY,ST):: IF ST=0 THEN 240 ELSE
    IF KEY=74 THEN GOSUB 1050
250 CALL CLEAR :: FOR BU=1 TO 50 ::
    CALL SOUND(-50,880+10*BU,3):: NEXT BU
260 DISPLAY AT(2,2):"SPIELERANZAHL [p,q,r]?" ::
    CALL KEY(3,KEY,ST):: IF ST=0 OR KEY<48 OR KEY>50
    THEN 260 ELSE G=KEY-48
270 CALL SOUND(1000,-2,1)
280 IF G<2 THEN CN$(2)=" TI 99" ::
    IF G<1 THEN CN$(1)=" 99pp" :: GOTO 340
290 FOR N=1 TO G
300 DISPLAY AT(4*G,2):"NAME VON SPIELER ";
    CHR$(112+N);" ?" ::
    ACCEPT AT(4*G+2,4)VALIDATE(UALPHA):CN$(N):: L=0
310 CN$(N)=SEG$(CN$(N),1,8)
320 NEXT N
330 CALL SOUND(600,440,2):: CALL SOUND(300,220,5)
340 CALL CLEAR :: GOSUB 680 :: IF RND<.5 THEN K=1 ::
    GOTO 370
350 K=2 :: I=-1
360 REM HAUPTPROGRAMM
370 CALL HCHAR(20,1,32,96)
380 L=L+1 :: S=1-(K=1)
390 IF K>10 THEN K=10 ELSE IF K<1 THEN K=1
400 A=H(K):: V3=(L>2)
410 DISPLAY AT(20,4):CN$(K);"'S ZUG:" ::
    IF K=2 AND G<2 THEN 1290
420 IF K=1 AND G<1 THEN 1290
430 DISPLAY AT(22,4):"ZEILE?"
440 CALL KEY(3,KEY,ST)::
    IF ST=0 OR KEY<49 OR KEY>54 THEN 440
450 Y=KEY-48 :: N=KEY+64 ::
    DISPLAY AT(22,11):CHR$(N)
460 DISPLAY AT(22,14):"SPALTE?"
470 CALL KEY(3,KEY,ST)::
    IF ST=0 OR KEY<65 OR KEY>70 THEN 470
480 X=KEY-64 :: DISPLAY AT(22,22):CHR$(KEY)::
    IF X=-44 THEN 530
490 GOSUB 1670
500 IF O(Y,X)=K OR B(Y,X)=0 THEN 540
510 DISPLAY AT(24,1)BEEP:"VERBOTENE POSITION
     - NOCHMAL"
```

TI 99/4A ***Erweitern Sie Ihr System***

- *volle Kompatibilität zum TI 99/4A*
- *softwaremässige Anpassung*
- *alle Komponenten einzeln funktionsfähig*
- *Erweiterung durch einfaches Aneinanderstecken*

mit System!

TI 99/4A

Nutzen Sie den Telefonkontakt und fordern Sie Unterlagen an: **040 - 668 22 56**

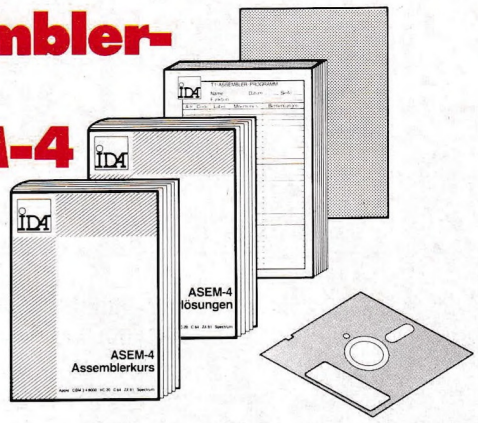
Schnittstellen - Speichererweiterungen - Disketten-Steuerungen - Disk-Laufwerke

atronic Bunke & Vernunft oHG
Postfach 700 124
2000 Hamburg 70

```
520 CALL SOUND(300,440,3):: CALL SOUND(900,220,3,
530 CALL HCHAR(20,1,32,160):: GOTO 410
540 B(Y,X)=B(Y,X)+1 :: O(Y,X)=K :: F=1 :: GOSUB 800 ::
    F=0 :: IF B(Y,X)=1 THEN D(K)=D(K)+1
550 Z=1 :: IF B(Y,X)=W(Y,X)THEN GOSUB 870
560 IF Z<>1 THEN 600
570 IF K=1 THEN K=2 :: GOTO 370
580 K=1 :: GOTO 370
590 REM SPIELENDE
600 CALL HCHAR(20,1,32,96)::
    DISPLAY AT(22,3):CN$(K);" HAT GEWONNEN!"
610 FOR BU=1 TO 3 :: CALL SOUND(100,500,1)::
    CALL SOUND(200,800,1):: NEXT BU
620 V(K)=V(K)+1 :: GOSUB 800 :: I=(K=2):: V3=0
630 FOR BU=1 TO 50 :: CALL SOUND(-50,880+10*BU,4)::
    NEXT BU
640 CALL KEY(3,KEY,ST)::
    IF ST=0 THEN 640 ELSE IF KEY=78 THEN CALL CLEAR ::
    STOP
650 CALL CLEAR :: FOR N=1 TO 6 :: FOR J=1 TO 6 ::
    B(N,J)=0 :: O(N,J)=0 :: NEXT J :: NEXT N ::
    D(1)=0 :: D(2)=0
660 L=0 :: I=NOT I :: GOSUB 680 :: K=1-I :: GOTO 370
670 REM SUB - SPIELBRETT-AUFBAU
680 DISPLAY AT(1,4):"*EIMER-KETTE*" ::
    DISPLAY AT(2,2):"abbbbbbbbbbbbbbbd" ::
    DISPLAY AT(3,2):"e  A B C D E F  e"
690 DISPLAY AT(4,2):"e abcbcbcbcbcbd e"
700 DISPLAY AT(5,2):"eqe e e e e e eqe"
710 FOR V5=1 TO 5 ::
    DISPLAY AT(4+2*V5,2):"e fbgbgbgbgbgbh e"
720 DISPLAY AT(5+2*V5,2):"e";CHR$(V5+113);
    "e e e e e e e";CHR$(V5+113);"e" :: NEXT V5
730 DISPLAY AT(16,2):"e ibjbjbjbjbjbk e"
740 DISPLAY AT(17,2):"e  A B C D E F  e" ::
    DISPLAY AT(18,2):"jbbbbbbbbbbbbbbbk"
750 DISPLAY AT(8,19):"SPIELSTAND" ::
    DISPLAY AT(10,19):CN$(1)::
    DISPLAY AT(12,19):CN$(2)
760 DISPLAY AT(10,28):CHR$(ASC(STR$(V(1)))):
    DISPLAY AT(12,28):CHR$(ASC(STR$(V(2))))
770 GOSUB 1660
780 RETURN
790 REM SUB - STEIN SETZEN
800 FOR N=1 TO 6 :: FOR J=1 TO 6
810 IF B(N,J)=0 THEN CALL HCHAR(2*N+3,2*J+5,32)::
    GOTO 840
820 IF O(N,J)=2 THEN VO=80
830 CALL HCHAR(2*N+3,2*J+5,ASC(STR$(B(N,J)))+VO)::
    VO=0 :: CALL SOUND(-100,-8,1)
840 NEXT J :: NEXT N
850 RETURN
860 REM SUB - UEBERLAUF
870 Y=1 :: X=1
880 IF B(Y,X)>=W(Y,X)THEN 920
890 X=X+1 :: IF X<=6 THEN 880
900 Y=Y+1 :: X=1 :: IF Y<=6 THEN 880
910 RETURN
920 CALL SOUND(-200,220,8,230,8,110,6,-7,10)::
    CALL HCHAR(3+Y*2,2*X+5,136)
930 CALL SOUND(200,240,6,230,8,110,4,-7,6)::
    CALL HCHAR(3+Y*2,2*X+5,137)
940 CALL SOUND(-300,260,4,800,0,300,2,-7,3)::
    CALL HCHAR(3+Y*2,2*X+5,138)
950 IF Y>1 THEN Q=Y-1 :: B(Q,X)=B(Q,X)+1 :: U=X ::
    GOSUB 1620
960 IF X>1 THEN U=X-1 :: B(Y,U)=B(Y,U)+1 :: Q=Y ::
    GOSUB 1620
970 IF Y<6 THEN Q=Y+1 :: B(Q,X)=B(Q,X)+1 :: U=X ::
    GOSUB 1620
980 IF X<6 THEN U=X+1 :: B(Y,U)=B(Y,U)+1 :: Q=Y ::
    GOSUB 1620
990 B(Y,X)=B(Y,X)-W(Y,X):: O(Y,X)=K
1000 IF B(Y,X)=0 THEN O(Y,X)=0 ::
    D(K)=D(K)-1
1010 F=1 :: GOSUB 800 :: F=0
1020 Z=1 ::
    IF (D(1)+D(2)>1)AND(D(1)=0 OR D(2)=0)THEN Z=0 ::
    RETURN
1030 GOTO 870
1040 REM SUB - INSTRUKTIONEN
1050 IO=1 :: CALL CLEAR
1060 DISPLAY AT(2,5):"eSPIELANLEITUNGe" ::
    DISPLAY AT(1,5):"abbbbbbbbbbbbbbd"
1070 DISPLAY AT(3,5):"ibbbbbbbbbbbbbk"
1080 DISPLAY AT(5,1):"BEI DIESEM SPIEL WERDEN AUF" ::
    DISPLAY AT(7,1):"DEM SPIEL BRETT ABWECHSELND"
1090 DISPLAY AT(9,1):"STEINE GESETZT. DABEI DARF" ::
    DISPLAY AT(11,1):"MAN NUR LEERE ODER EIGENE"
1100 DISPLAY AT(13,1):"FELDER BELEGEN. WIRD DIE" ::
    DISPLAY AT(15,1):"MAXIMALE BELEGZAHL UEBER-"
1110 DISPLAY AT(17,1):"SCHRITTEN, LAEUFT DAS FELD" ::
    DISPLAY AT(19,1):"UEBER UND ERGIESST SEINEN"
1120 DISPLAY AT(21,1):"INHALT AUF DIE NACHBARFELDER"
1130 GOSUB 1250 :: CN$(1)="SPp" :: CN$(2)="SPq"
1140 FOR N=1 TO 6 :: FOR J=1 TO 6 :: O(N,J)=0 ::
    B(N,J)=0 :: NEXT J :: NEXT N :: O(2,1)=1 ::
    O(4,3)=2
1150 B(2,1)=2 :: B(4,3)=3 :: GOSUB 680 :: GOSUB 800 ::
    DISPLAY AT(4,19):"SO IST DAS" ::
    DISPLAY AT(6,19):"FELD, WENN"
1160 DISPLAY AT(8,19):"r STEINE" ::
    DISPLAY AT(10,19):"AUF rA"
1170 DISPLAY AT(12,19):"UND s AUF" ::
    DISPLAY AT(14,19):"FELD tC" ::
    DISPLAY AT (16,19):"LIEGEN."
1180 GOSUB 1250
1190 FOR N=1 TO 6 :: FOR K=1 TO 6 ::
    B(N,K)=W(N,K)-1 :: O(N,K)=1 :: NEXT K :: NEXT N
    :: GOSUB 680 :: GOSUB 800
1200 DISPLAY AT(4,19):"DIES IST" ::
    DISPLAY AT(6,19):"DIE" ::
    DISPLAY AT(8,19):"MAXIMALE"
1210 DISPLAY AT(10,19):"BELEGUNG" ::
    DISPLAY AT(12,19):"ALLER" ::
    DISPLAY AT(14,19):"FELDER."
1220 GOSUB 1250
1230 DISPLAY AT(12,4):"*** UND JETZT GEHT^S LOS ***"
1240 FOR N=1 TO 6 :: FOR K=1 TO 6 :: B(N,K)=0 ::
    O(N,K)=0 :: NEXT K :: NEXT N :: IO=0 ::
    CN$(1)="" :: CN$(2)=""
1250 REM SUB - INSTRUKTION SEITE WECHSELN
1260 DISPLAY AT(24,5):"DRUECKE EINE TASTE"
1270 CALL KEY(3,KEY,ST)::
    IF ST=0 THEN 1270 ELSE CALL CLEAR :: RETURN
1280 REM AUTOMATIK
1290 Q=0 :: U=Q :: FOR N=1 TO 6 :: FOR J=1 TO 6
1300 P(N,J)=0 ::
    IF O(N,J)=K AND B(N,J)=W(N,J)-1 THEN P(N,J)=1 ::
    Q=Q+1
1310 IF O(N,J)<>S THEN U=U+1
1320 NEXT J :: NEXT N :: IF Q=0 THEN 1470
1330 FOR N=1 TO 6 :: FOR J=1 TO 6 ::
    IF P(N,J)=0 THEN 1420
1340 OX=J :: FOR PX=N-1 TO 1 STEP -1 :: GOSUB 1590 ::
    IF B(PX,J)<>W(PX,J)-1 THEN PX=1
1350 NEXT PX :: IF N>5 THEN PX=1 :: GOTO 1380
1360 FOR PX=N+1 TO 6 :: GOSUB 1590 ::
    IF B(PX,J)<>W(PX,J)-1 THEN PX=6
```

```
1370 NEXT PX
1380 PX=N :: FOR OX=J TO 1 STEP -1 :: GOSUB 1590 ::
     IF B(N,OX)<>W(N,OX)-1 THEN OX=1
1390 NEXT OX :: IF J>5 THEN 1420
1400 FOR OX=J+1 TO 6 :: GOSUB 1590 ::
     IF B(N,OX)<>W(N,OX)-1 THEN OX=6
1410 NEXT OX
1420 NEXT J :: NEXT N
1430 M=0 :: FOR PX=1 TO 6 :: FOR OX=1 TO 6
1440 IF P(PX,OX)>M THEN M=P(PX,OX):: N=PX :: J=OX
1450 NEXT OX :: NEXT PX ::
     IF Q<>1 OR((N=1 OR N=6)AND(J=1 OR J=6)AND RN(2)=1)
     THEN 1570
1460 IF Q=1 AND P(N,J)>50 THEN 1570
1470 B3=0 :: IF RN(3)=1 OR U>33 THEN 1500
1480 FOR PX=1 TO 6 :: FOR OX=1 TO 6 ::
     IF O(PX,OX)=K THEN N=PX :: J=OX :: PX=6 :: OX=6
10-5 IT1H N1 RR IT1H Q1 RR YNHN 2625
2655 I{SI*7|RR V{SI*7|RR LC N*IWV|{E HOTI 2655
2625 S{5 RR CNS Q1{I$2 HN I$*I)7|RR
     LC N*Q1WV|)]E HOTI 2645
2635 LC F*PX,J)=W(PX,J)-1 THEN R=1
1530 NEXT PX :: FOR OX=J-1 TO J-(J<6)::
     IF O(N,OX)<>S THEN 1550
1540 IF B(N,OX)=W(N,OX)-1 THEN R=1
1550 NEXT OX :: IF R THEN B3=B3+1 ::
     IF B3<U THEN 1500
1560 GOTO 1570
1570 DISPLAY AT(20,18):CHR$(112+N);CHR$(J+64)::
     GOSUB 1670 :: X=J :: Y=N :: GOTO 540
1580 REM SUB - PUNKTE
1590 P(N,J)=P(N,J)+1 ::
     IF O(PX,OX)=S THEN P(N,J)=P(N,J)+50
1600 RETURN
1610 REM SUB - ERHOEHUNG NACHBARFELD
1620 IF O(Q,U)=S THEN D(S)=D(S)-1
1630 IF O(Q,U)<>K THEN D(K)=D(K)+1
1640 O(Q,U)=K :: RETURN
1650 REM SUB - MUSIK
1660 RESTORE 1740 :: FOR I=1 TO 31 ::
     READ T1,T2,T3,L :: CALL SOUND(L,T1,0,T2,6,T3,5)::
     NEXT I :: RESTORE :: RETURN
1670 CALL SOUND(150,240,2):: CALL SOUND(150,360,2)::
     CALL SOUND(300,120,2):: RETURN
1680 REM DATEN - ARRAY
1690 DATA 2,3,3,3,3,2
1700 REM DATEN - CHARACTER
1710 DATA 0000001F1F181818,000000FFFF,000000FFFF181818,
     000000F8F8181818,1818181818181818,
     1818181F1F181818,181818FFFF181818,
     181818F8F8181818
1720 DATA 1818181F1F,181818FFFF,181818F8F8
1730 REM DATEN - MUSIK
1740 DATA 740,496,147,400,494,494,147,200,587,494,147,
     200,740,494,294,200,587,494,294,200,494,494,294,
     400
1750 DATA 784,494,330,800,659,440,277,400,440,440,277,
     200,554,440,277,200,659,440,277,200,554,440,277,
     200
1760 DATA 440,440,277,400,740,440,294,800,659,494,196,
     200,740,494,196,200,659,494,196,200,587,494,196,
     200
1770 DATA 554,440,220,400,698,440,220,400,659,392,247,
     200,698,392,247,200,659,392,247,200,587,392,247,
     200
1780 DATA 440,392,220,400,587,349,220,400,554,330,220,
     200,587,330,220,200,659,392,220,200,587,392,220,
     200
1790 DATA 587,349,294,2400 ■ H.U.
```


Assembler-kurs ASEM-4



Mehr rausholen aus dem »Micro« mit schnelleren und leistungsfähigeren Maschinen-Programmen. —
Lernen Sie Assembler-Programmierung und Computer-Aufbau verstehen. **In direktem Bildschirm-Dialog** und begleitet von zahlreichen Übungsaufgaben, wird der Stoff in leicht verständlicher Form vorgestellt.

Zum Kurs gehören:

- **2 deutsche Handbücher (ca. 400 Seiten),**
- **Lösungsblätter und Referenzkarte,**
- **Software auf Diskette oder Cassette.**

Wählen Sie für Ihr System passend:

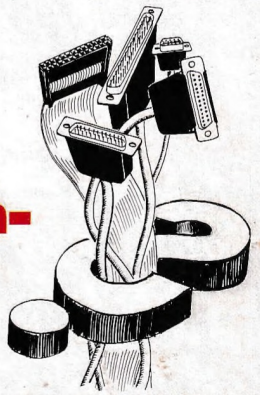
Cassette:	
CBM 3/40xx	98,-
CBM 80xx	98,-
VC-20 (16K)	98,-
C 64	98,-
ZX-81 (32K)	98,-
Spectrum (48K)	98,-
Genie I, II, III	98,-
Colour-Genie	98,-
TRS-80 I	98,-
ATARI	98,-
TI-99	98,-

Diskette:	
CBM 3/40xx (4040)	98,-
CBM 80xx (8050)	98,-
VC-20 (16K, VC-1541)	98,-
C 64 (VC-1541)	98,-
Apple II, IIe	98,-
Genie I, II, III	98,-
TRS-80 I (35-Track)	98,-
Colour-Genie	98,-
MS-DOS (IBM-PC u.v.a.)	148,-

CP/M, je nach Rechner-Typ: auf Diskette oder auf Cassette mit Cass.-Interface über V24-Schnittstelle 148,-

. . . andere Systeme auf Anfrage!

Schnittstellen-Probleme?



Schnittstellen-Probleme schnell erkennen und beheben! Für viele Anwendungsfälle haben wir den passenden Interface-Tester oder Adapter:

RS 232 / V24 Schnittstellen-Tester, unterbrechen, kreuzen und überbrücken aller Datenleitungen (DIP-Switches / Steckfelder), Signal-Anzeige durch LEDs, BNC-Buchsen für Oszilloscope-Anschluß, ein Gerät für die Praxis. Komplett mit Kabeln und deutscher Anleitung **DM 548,-**

RS-232 / V24 Mini-Adapter, löst das bekannte Problem unterschiedlicher Pin-Belegungen; unterbrechen, vertauschen und brücken aller Datenleitungen, Anpassung an spez. Peripherie-Geräte in Sekunden.
DM 132,- (1 St.) **DM 114,-** (ab 3 St.) **DM 99,-** (ab 10 St.)

Adapter für andere Kabel-Systeme auf Anfrage!

Alle Preise incl. MwSt. **Versand per NN**

ING.-BÜRO WILKE

Postfach 1727 · 5100 Aachen 1 · Tel. 0241 / 3 06 81 + 87 02 08

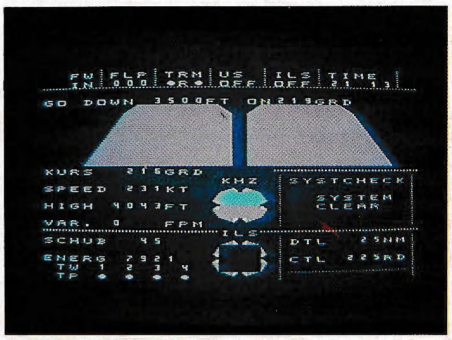
SOFTWARE IM TEST

Flugsimulator

Diese Cassette beinhaltet einen Flugsimulator in EXTENDED BASIC und TI BASIC und wird mit Joysticks und Tastatur bedient.

Die Flugeigenschaften des simulierten Flugzeugs entsprechen in etwa denen eines Jumbo Jets: Der Flugsimulator reagiert auf Joystickanweisungen nur sehr langsam. Nun soll versucht werden, den Jet aus einer Höhe von 1300 Metern sicher auf dem Boden zu landen. Dabei müssen Landeklappen und Fahrwerk ausgefahren und der Instrumenten-Lande-System-Empfänger (ILSE) eingeschaltet werden. Anweisungen, wie zu fliegen ist, erhält man vom Landekurssender des Towers.

Auf dem Bildschirm sieht man dann schließlich das Cockpit des Flugzeugs,



einen künstlichen Horizont, einen Entfernungsanzeiger, den geflogenen Kompaßkurs, die momentane Flughöhe, die Geschwindigkeit und noch eine ganze Menge weiterer Anzeigen.

Nachteilig an diesem Spiel ist vor allem die fehlende Graphik. Die Mühe eines Landebahnentwurfs haben sich die Spielautoren einfach gespart. Nicht einmal Wolken tauchen auf. Übrig bleibt ein Spiel, das durch seine Komplexität, aber keineswegs durch seine Graphik oder seine Schnelligkeit Spielspaß vermittelt. Wer keine Ausdauer hat, sollte sich dieses Spiel allerdings nicht kaufen, denn Ausdauer braucht man, um die durch die Praxisnähe bedingte anfängliche Frustration zu überstehen, um nicht die Cassette in die Ecke zu schmeißen und sie nie wieder auch nur eines einzigen Blickes zu würdigen.

Graphik:

Sound:

Spielidee:

Testergebnis:

Sympathie

Unseres Wissens haben wir nie einen Eid abgelegt, Bücher objektiv zu beurteilen. Schließlich sind wir selbst User und es wäre schon komisch, wenn wir das vergessen würden. Und so bricht uns kein Zacken aus der Krone, wenn wir sagen: dieses Buch für den TI 99/4A gefällt uns einfach, weil der Schreibstil witzig und symphatisch ist. Echter Lesespaß.

Inhaltlich bietet es neben kurzen Abschnitten über "Farben und Grafik", "Töne und Akkorde" sowie "Kniffe und Tricks" eine Menge ausgefuchster Spiele. 192 Seiten Listings mit Einleitung und Erklärung zu jedem Spiel (den Autor hat die lockere Schreibe bis ans Ende des Buches zum Glück nicht verlassen). Die Listings sind auf 28 Zeichen pro Zeile formatiert, was günstig für das Abtippen ist. Sehr gut gefiel uns, daß parallel zu den Listings laufende Dokumentation steht.

Guido Pahlberg
IWT-Verlag München
ISBN 3-88322-045-0
217 Seiten
39,— DM

Als Spiele finden sich hier, um nur einige zu nennen: FORMEL-I, LABYRINTH-LAUF, SUPERMIND (nun, es hat ja keiner gesagt, daß alles neu ist), SENSOCOMP, SQUASH und TENNIS, ELIZA als Beispiel von AI (Künstlicher Intelligenz) und SCHWARZES MEMORY zur Illustration von AD = Artificial Dumbness (Künstlicher Dummheit). Und vieles mehr.

Zu jedem Listing gibt es Anregungen, das Programm zu ändern, getreu dem Motto, daß zumindest der Fortgeschrittene angeregt werde, seiner Phantasie freien Lauf zu lassen.

"Nur zu, Freunde! Diesen kleinen Riesen TI 99/4A gilt es, an seine Grenzen zu führen. Viel Spaß". Dem können wir nichts hinzufügen.

TI 99/4A Farbe, Grafik, Ton, Spiele (ist als Programm-Cassette ebenfalls lieferbar)

ThoK

Endlich sind die ersten Atari-Moduln auch für den TI99/4A zu bekommen. Die Firma Atari hat uns die beiden ersten adaptierten Steck-Moduln

ZU TI TELEFONBUCH
Heft 5/84, Seite 76

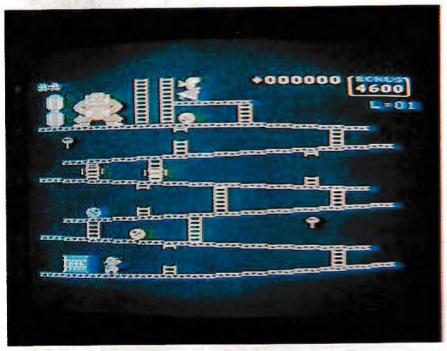
Listing-Zeile 130 ist nur mit Trick in den Rechner zu bringen!

Nach der Meldung "Zeile voll" (akustisches Signal) ENTER drücken, Editier-Modus verlassen, Zeile 130 neu editieren (z.B. mit NUM 130).

Eingabe fortsetzen (natürlich am Ende der bereits eingegebenen Zeile), bis erneutes Signal kommt, dann den Ablauf wiederholen.

DIE ZEILE LÄSST SICH NUR SO EINGEBEN!

zugeschickt: Donkey Kong und Defender. Beide Spiele dürften aus Spielhallen oder von Atari- bzw. Commodore-Rechnern vielen von euch bekannt sein. Für alle, die noch nichts davon gehört haben, hier eine kurze Beschreibung:



Donkey Kong

Marios Freundin ist von dem furchterregenden Monster-Affen Donkey Kong gefangen und auf ein Baugerüst verschleppt worden. Mario muß jetzt zu seiner Freundin hinaufklettern, um sie zu retten. Aber das gefällt Donkey Kong gar nicht, und deshalb wirft er mit allerlei Gegenständen um sich: mit Fässern, Sandeimern und Flammwesen, die den armen Mario auf seinem Weg nach oben attackieren. Jedesmal, wenn es ihm gelingt, über eines dieser Hindernisse hinwegzuspringen, werden ihm zur Belohnung Punkte gutgeschrieben.

Die bekannte, ansprechende Graphik des Spiels ist von Atari erfolgreich auf den TI übertragen worden. Nur der Sound, besonders die Start-Melodie, kann einem schnell auf die Nerven gehen. Aber da helfen die — ebenfalls bekannten — Möglichkeiten eines Lautstärke-Reglers.

Graphik:

Sound:

Spielidee:

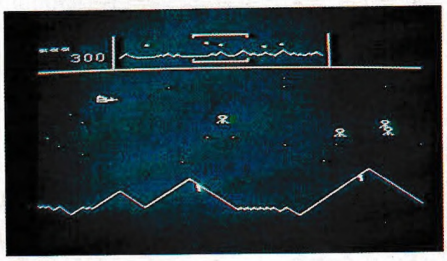
Testergebnis:

Defender

Eines der ältesten 'Baller-Spiele' überhaupt. Einziges Ziel: schießen, schießen, schießen. Und natürlich treffen! Wie es in der Anleitung steht: die „totale Vernichtung" des Gegners wird angestrebt. Der ist aber auch nicht faul, versucht er doch, mit Spezial-Schiffen die 'Humanoiden' des Verteidigers zu kidnappen und als Mutanten auf seiner Seite wieder in den Kampf zu schicken. Mal ganz davon abgesehen, daß die Angreifer auf den Verteidiger blindwütig schießen, was das Zeug hält.

Hat man erfolgreich alle Feinde vernichtet — rollt schon die nächste Angriffswelle an.

Die sowieso recht einfache Graphik ist problemlos an das TI-System adaptiert worden. Die hektischen Bewegungen des Defender-Raumschiffes gehen



aber reichlich auf die Augen. Großartige Variationsmöglichkeiten gibt es nicht. Auch der Sound ist nach einfachstem Muster gestrickt: man hat sich auf das übliche 'Weltraum-Rauschen und -Krachen' beschränkt.

Jedes dieser beiden Spiele kostet ca. 100,— Mark. Dafür bekommt man jeweils das Steckmodul mit einer ausführlichen deutschen Anleitung, in der sowohl die Inbetriebnahme des Moduls als auch das Spiel selbst genau beschrieben werden. Kleine Fehler, wie z.B. die Anweisung, die Joysticks an der rechten Seite des Rechners anzuschließen, muß man großzügig überlesen.

Übrigens: beide Spiele lassen sich nur mit Joysticks betreiben! Dafür allerdings auch zu zweit. Bleibt nur das Problem, wo man vernünftige Steuerknüppel herbekommt, sofern man keine TI-Original-Joysticks besitzt.

Einen Adapter für TI-fremde Joysticks haben wir in dieser Ausgabe beschrieben. Aber trotz dieser Möglichkeit zur Selbsthilfe ist es ein schwaches Bild, daß professionelle Software nach wie vor 'Peripherien' verlangt — und seien es nur Joysticks.

Graphik:

Sound:

Spielidee:

Testergebnis:

to be AND OR NOT to be!

Präzise Aussagen sind nötig, wenn man mit dem Computer plauschen will. Na und? Oder oder? Oder nicht?

In dieser Ausgabe von COMPUTER Praxis wollen wir die TI-User mit der Möglichkeit vertraut machen, logische Funktionen — wie AND, OR und NOT — auch ohne Zuhilfenahme von Ex-Basic zu realisieren.

Was sind logische Funktionen? Sie ermöglichen dem Programmierer, mit IF ... THEN-Befehlen bedingte, d.h. von ganz bestimmten Bedingungen abhängige, Programmsprünge zu anderen Programmteilen zu machen. Bei IF...THEN-Befehlen erfolgt ein Sprung nur dann, wenn der zwischen IF und THEN stehende Ausdruck "wahr" ist. Dieser Ausdruck kann aus einer Vielzahl von logischen Verknüpfungen bestehen. Die wichtigsten logischen Verknüpfungselemente sind AND, OR und NOT.

Sag' die Wahrheit!

Wenden wir uns zunächst dem AND zu. Werden zwei Elemente, z.B. A und B, mit AND verknüpft, so ist das Ergebnis nur dann wahr, wenn sowohl A als auch B wahr ist. Das klingt kompliziert, ist es aber gar nicht, wie ihr gleich an einem einfachen Beispiel sehen könnt. Nehmen wir einmal an, A und B wären zwei Schalter und das wahre Ergebnis soll eine brennende Glühlampe sein. Das bedeutet für die Verknüpfung AND, daß die Lampe dann leuchten muß, wenn sowohl Schalter A als auch Schalter B eingeschaltet, sprich wahr sind. Aus Abb.1 könnt ihr ersehen, daß die Schalter hintereinander geschaltet werden müssen, um unsere AND-Schaltung zu erhalten. Wird nur ein Schalter — entweder A oder B betätigt —, so fließt kein Strom zur Lampe. Das Gleiche geschieht, wenn wir gar keinen Schalter drücken. Nur bei Betätigung beider Schalter kann Strom zu unserer Lampe fließen. Das können wir auch mit einer sogenannten Wahrheitstabelle allgemeiner darstellen. Hier könnt ihr in Abb.2 sehen, daß unser Ergebnis C nur dann wahr ist, wenn auch A und B wahr sind. In einer solchen Tabelle wird, wie ihr sicherlich schon bemerkt habt, für jede Verküpfungsmöglichkeit das entsprechende Ergebnis angegeben.

Abb. 2

Ereignis		Ergebnis
A	B	C
falsch	falsch	falsch
wahr	falsch	falsch
falsch	wahr	falsch
wahr	wahr	wahr

Abb. 1

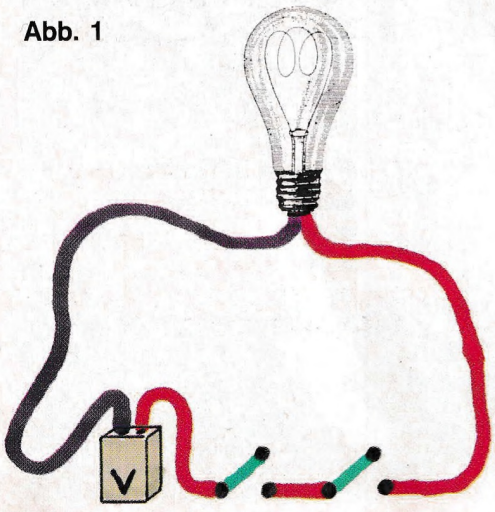
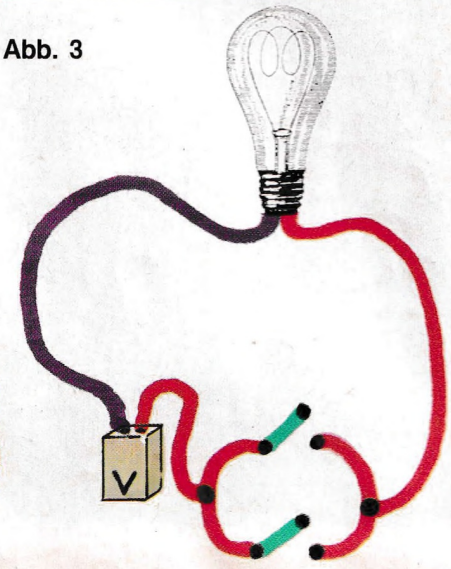


Abb. 3



Die OR-Verkünpfung wird zu Recht auch als Entweder-Oder-Verküpfung bezeichnet. D.h. werden zwei Ereignisse A und B durch OR verknüpft, so ist das Ergebnis C dann wahr, wenn entweder A oder B oder beide wahr sind. Für unser 'Lampenbeispiel' sieht die Schaltung wie in Abb. 3 gezeigt aus. Wird einer der parallel geschalteten Schalter oder beide betätigt, fließt der Strom wieder zu unserer Lampe. Eine Wahrheitstabelle für OR zeigt Abb.4.

Abb. 4

Ereignis		Ergebnis
A	B	C
falsch	falsch	falsch
wahr	falsch	wahr
falsch	wahr	wahr
wahr	wahr	wahr

Die NOT-Verknüpfung stellt alles im wahrsten Sinne des Wortes auf den Kopf: Ist ein Ereignis A wahr, so soll das Ergebnis C falsch sein und umgekehrt. Wird also ein Schalter A betätigt, muß unsere brennende Lampe ausgehen. Eine Schaltung dafür solltet ihr euch einmal selbst überlegen. Die Wahrheitstabelle ist in Abb.5 zu finden.

Abb. 5

Ereignis	Ergebnis
A	C
falsch	wahr
wahr	falsch

Nach soviel Theorie fragt ihr euch sicherlich: Wofür brauchen wir eigentlich diese logischen Verknüpfungen in unserem Programm? Das ist ganz einfach zu beantworten. Nehmen wir einmal an, ihr wollt ein Spiel programmieren, in dem es für den Spieler fünf Männchen

gibt, die von mehreren Monstern verfolgt werden. Sinn des Spiels ist es, sich möglichst lange nicht erwischen zu lassen. Nun taucht beim Programmieren sicher die Frage auf, wann ist das Spiel beendet? Doch, sobald das fünfte Männchen von einem Monster erwischt wird! Um den Sprung zum Programmende zu programmieren, müssen genau zwei Ereignisse eintreten, die den Sprung vollziehen. Erstens: Mein Männchen muß erwischt werden, und zweitens: Es muß mein letztes Männchen sein. Ganz Schlaue werden jetzt bemerkt haben, daß es sich hier um eine AND-Verknüpfung handeln muß. Denn sowohl Ereignis A — mein Männchen wird geschnappt — als auch Ereignis B — es ist mein letztes Männchen — müssen erfüllt, sprich wahr sein, damit das Ergebnis C — Sprung zum Ende des Programms — auch wahr wird.

Auch eine OR-Verknüpfung ist in unserem Programm denkbar. Es kann nämlich eintreten, daß unser Männchen entweder von einem oder von mehreren Monstern gleichzeitig einkassiert wird. In jedem Fall gilt unser Männchen als erwischt, und ein neues Männchen muß ins Spiel gebracht werden. Der Sprung

FUNKTIONIERT OHNE FRUST: KOMFORTABLE SOFTWARE VON DYNAMICS.

Weil es nicht nur Computer-Freaks gibt, bietet Dynamics anwender-freundliche Software – komfortabel, menuegesteuert über den Bildschirm.

Wer also auf seinem Commodore C64, dem TI99/4A oder den Atari 400/800- oder XL-Modellen nicht lange programmieren will oder null Bock auf Listings hat, der geht ins Fachgeschäft oder Kaufhaus.
Dort gibt es das Dynamics **„Adressen-Archiv"**, das **„Video-"**, **„Bücher-"** und **„Schallplatten-Archiv"**: Programme, die Ordnung halten – einfach und flink. Da gibt es neu den **„Vokabel-Trainer"**, der Lernen zum Vergnügen macht. Fragen Sie danach!

Wer mehr will, sollte sich die Dynamics Arbeits-Programme zeigen lassen. Aber auch **„Word-Proc"**, **„Tool Pack I** und **II"**, **„Calculator"** oder **„Data-Bank"** bleiben immer anwender-freundlich.

Gut verständliche Bedienungsanleitungen (keine Wälzer!), logischer Aufbau und einfache Anwendung garantieren, daß computern mit Dynamics Spaß macht!

Also: auf ins Fachgeschäft, hin zum Kaufhaus. Weil Dynamics Software einfach komfortabel ist.
Ausprobieren!

Dynamics-Software gibt es auf Computer-Cassette oder Diskette.

DYNAMICS®

COMPUTER-SOFTWARE UND COMPUTER-ZUBEHÖR.

FÜR ATARI 400/800 UND 600 XL/800 XL, TEXAS INSTRUMENTS TI 99/4A UND COMMODORE C 64.

Dynamics Marketing GmbH,
Große Bäckerstraße 11,
2000 Hamburg 1.

wird demnach dann wahr, wenn unser Spieler entweder von einem oder mehreren geschnappt wird.

Eine NOT-Bedingung hätten wir z.B. dann vorliegen, wenn ein Sprung zum weiteren Programmablauf immer dann erfolgen soll, falls unser kräftiger Spieler nicht auf frischer Tat ertappt wurde.

Das Ex-BASIC und die meisten anderen BASICs erlauben die direkte Benutzung der Vokabeln AND, OR und NOT. Das ist beim TI-BASIC nicht so. Deshalb müssen wir uns an dieser Stelle etwas einfallen lassen.

Zuallererst wollen wir uns ansehen, was beim TI "wahr" und "falsch" bedeutet. Dazu solltet ihr einmal nachstehendes Listing eingeben.

```
100 INPUT A
110 IF A THEN 200
120 GOTO 300
200 PRINT "WAHR"
210 GOTO 100
300 PRINT "FALSCH"
310 GOTO 100
```

Bei diesem kleinen Programm könnt ihr verschiedene Werte für A eingeben und bekommt je nach Eingabe eine wahre Aussage oder eine falsche. Ihr habt sicherlich bemerkt, daß alles ungleich 0 "wahr" ist, also nur 0 für "falsch" und alle andere Werte — ob positiv oder negativ — in jedem Fall für "wahr" stehen.

Monströsitäten

Nun wollen wir zu unserem Männchenbeispiel zurückkehren. Wir hatten festgestellt, daß unser Sprung zum Programmende dann ausgeführt werden soll, wenn wir erwischt wurden, und unser letzter Spieler der Betroffene war. Das kann man in EX-BASIC, wie in folgendem Listing ersichtlich, leicht bewerkstelligen:

```
100 IF (E=1) AND (M=5) THEN
    1000
110 REM HIER LAEUFT UNSER
    PROGRAMM WEITER
120 REM WENN WIR NICHT ER-
    WISCHT WURDEN ODER
130 REM WIR ZWAR ERWISCHT
    WURDEN ABER NOCH
140 REM NICHT MIT UNSEREM
    LETZTEN MAENNCHEN
1000 REM HIER BEGINNT UNSERE
     ENDE-ROUTINE
1010 REM UNSER LETZTES MAENN-
     CHEN WURDE ERWISCHT
```

Und zwar steht E für erwischt und wird vorher von uns immer dann auf eins gesetzt, wenn wir geschnappt wurden. M zeigt an, welches Männchen momentan im Spiel ist. Immer wenn E = 1 und(!) M = 5 ist, verzweigt unser Programm zur Zeile 1000. Nun kennt das normale TI-BASIC das Schlüsselwort "AND" aber nicht. Wir müssen uns also etwas einfallen lassen! Die Lösung ist ganz einfach. Folgende Programmzeile löst all unsere Probleme:

```
100 IF (E=1)*(M=5) THEN 1000
```

Statt des AND ein Multiplikationszeichen? Richtig! Die Gründe dafür sind sehr einfach zu verstehen. Tippt einmal PRINT 1 = 1, was sicherlich richtig ist, ein und ihr werdet eine -1 für "wahr" erhalten. Bei PRINT 1 = 3, das führt logischerweise zu einem "falschen" Ergebnis, erhaltet ihr den Wert 0. D.h., ist ein Vergleich richtig, liefert der TI uns eine -1, und ist er falsch, eine 0. Ist aber in unserer obigen Zeile ein Vergleich falsch, so ist dieser gleich 0. Eine Multiplikation mit einer Null ergibt unabhängig davon, womit ich sie multipliziere, immer eine Null als Ergebnis, und diese steht, wie wir gesehen haben, für "falsch". Nur wenn beide Vergleiche also richtig sind (= -1), wird unser Ergebnis auch "wahr" sein (-1 multipliziert mit -1 ergibt 1. 1 versteht unser TI aber auch als "wahr"). Der Vergleich mit unserer Wahrheitstabelle zeigt uns, daß wir hier anscheinend richtig programmiert haben.

Jetzt zur OR-Bedingung. Wir hatten in unserem Beispiel gesagt, daß wir dann verzweigen wollten, wenn uns eins oder mehrere Monster erwischt haben. Nehmen wir an, daß in unserem Programm für die drei Monster, die uns jagen, jeweils eine Abfrage erfolgt, ob sie uns ertappt haben. Ist das der Fall, dann wird für Monster 1 die Variable M1 gleich 1 gesetzt, für Monster 2 die Variable M2 usw. Unser EX-BASIC-Programm sieht unter diesen Voraussetzungen so aus:

```
100 IF (M1=1) OR (M2=1) OR
    (M3=1) THEN 1000
110 REM WIR WURDEN NICHT
    ERWISCHT UND
120 REM UNSER PROGRAMM
    LAEUFT WEITER
    .
    .
    .
1000 REM HIER BEGINNT DIE
     "WIR-WURDEN-ERWISCHT
     ROUTINE"
```

Wir sehen: Ist auch nur eine der Variablen M1 bis M3 gleich 1, springt unser Programm in Zeile 1000. In TI-BASIC müssen wir uns wieder behelfen. Wir nehmen statt der Multiplikation einfach die Addition, und schon funktioniert unser Befehl auch im normalen BASIC:

```
100 IF (M1=1)+(M2=1)+(M3=1)
    THEN 1000
```

Keine Not mit NOT

Das funktioniert folgendermaßen. Wird nur ein einzelner Vergleich wahr, so erhält dieser und damit der ganze Ausdruck den Wert -1. Ertappen uns mehrere Monster, erhalten wir die Werte -2 oder-3. -1 bis -3 sind jedoch alles "wahre" Werte. Nur wenn uns keiner erwischt, ist auch die Addition dieser "Nullen" eine Null und das ist eine "falsche" Aussage. Darum erfolgt in diesem Fall auch kein Sprung.

Last but not least die NOT-Verknüpfung. Diese wollen wir durch Probieren studieren:

```
100 INPUT A
110 IF (A=0)THEN 200
120 GOTO 300
200 PRINT "WAHR"
210 GOTO 100
300 PRINT "FALSCH"
310 GOTO 100
```

Geben wir für A "wahre" Werte (positive oder negative Zahlen) ein, ergibt das ein falsches Ergebnis. Wird A = 0 wie "falsch" gesetzt, bekommen wir "wahr" auf unseren Bildschirm. Warum? Wir vergleichen unsere Eingabewerte mit 0, ist dieser Vergleich "wahr", dann müssen wir vorher eine "falsche" Zahl (= 0) eingegeben haben und entsprechend umgekehrt.

Mit diesem Rüstzeug versehen, könnt ihr nun eure Programme hoffentlich besser gestalten. ■

J.P.P.

TI Sound-Check

Damit Monster brüllen, Motoren dröhnen und dem Sieger ein Tusch gespielt wird: Tonerzeugung auf dem TI 99/4A in BASIC

Da der TI 99/4A vier Tonerzeuger zur Verfügung hat, davon ist einer ein Rausch-Generator, ist man verlockt, ein Spiel mit dem richtigen Sound zu unterlegen.

Doch wie mühsam ist es, die beste Zusammenstellung von Frequenzen und Rauschen zu finden! COMPUTER Praxis, ein Fachblatt, wenn es um den guten Ton geht, bietet hier ein BASIC-Programm an, das euch via Joystick den "Sound checkt".

Zeile 100: Setzt die Arbeitsvariable FREQ gleich 1.

Zeile 110: Dimensioniert das Array FRE.

Zeile 200—380: Hier wird mit dem Joystick die Frequenz des jeweils geschalteten Generators verändert. Bei Druck auf den Feuerknopf wird der nächste Generator angewählt bzw. das Programm fortgesetzt.

Zeile 420—550: Wenn gewünscht, wird ein Rauschgenerator dazugeschaltet. Welches Rauschen gesetzt wird, bestimmt der Joystick. Per Feuerknopf wird das Programm fortgesetzt.

Zeile 590—700: Ein letzter Test. Oder auch nicht. Ihr könnt jetzt ganz gezielt einzelne Frequenzen verändern und dann den Gesamteindruck erneut testen. Die gewählten Frequenzen und der Rausch-Generator werden so ausgegeben, wie sie im CALL SOUND-Befehl eingesetzt werden müssen.

Zeile 740—950: Hier findet die eben beschriebene Änderung statt.

Ziele 990—1070: Diese Sub-Routine erzeugt den mehr oder weniger gewünschten Sound. Da die Tonlänge als negative Zahl (-4000) eingesetzt ist, wird der Ton während des ganzen Programmlaufs durchgehend wiedergegeben.

Die Steuerung der Frequenzen bzw. der Rausch-Arten erfolgt mit dem Joystick 1. Mit dem Feuerknopf kommt ihr in die jeweils nächste Arbeitsstufe. Wird ein Menü vorgegeben, so ist es gleich, ob Buchstaben groß oder klein eingegeben werden.

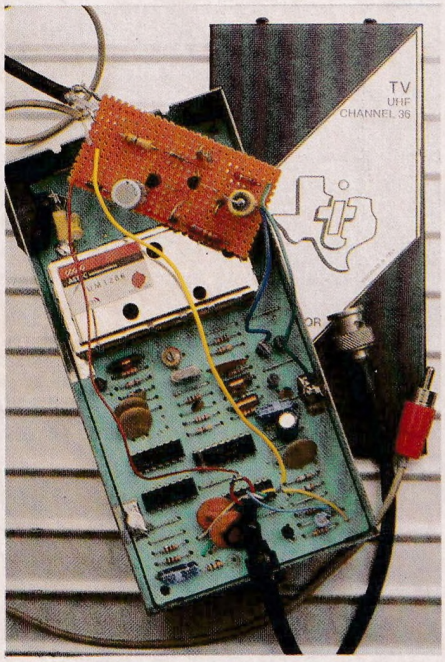
Mit diesem kleinen Hilfsprogramm könnt ihr euch eine Vielfalt verrückter Sounds erstellen. ■

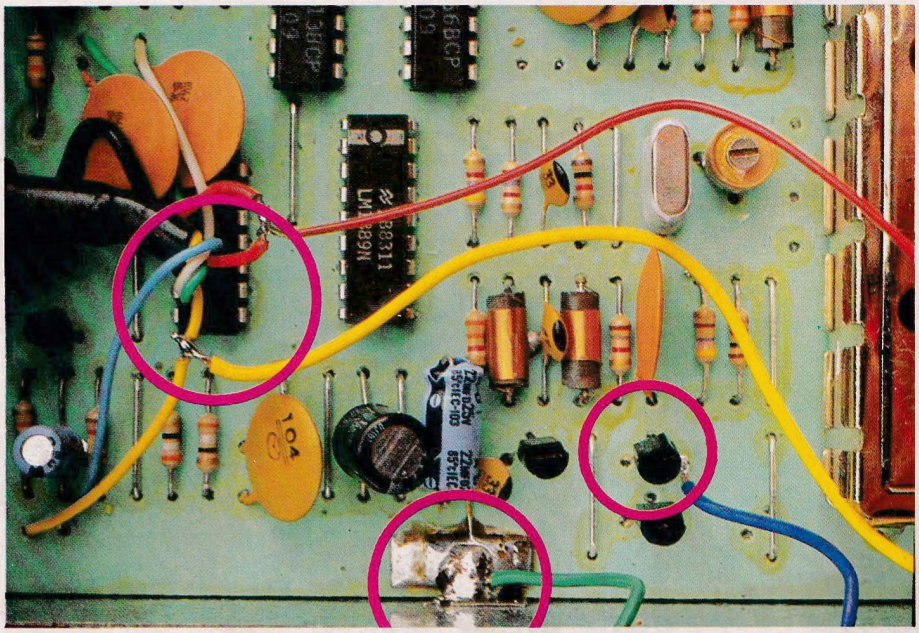
HU

```
100 FREQ=1
110 DIM FRE(3)
120 CALL CLEAR
130 PRINT TAB(10);"SOUND-CHECK": :
: : : :
140 FRE(FREQ)=110
150 PRINT "TON-GENERATOR";FREQ
160 GOSUB 990
170 REM
180 REM FREQUENZ-AENDERUNG
190 REM
200 CALL KEY(1,K,S)
200 CALL KEY(1,K,S)
210 IF S=0 THEN 300
220 IF FREQ<>3 THEN 280
230 PRINT : :
"RAUSCH-GENERATOR = 'J'"
240 GOSUB 990
250 CALL KEY(3,K,S)
260 IF S=0 THEN 240
270 IF K=74 THEN 420 ELSE 590
280 FREQ=FREQ+1
290 GOTO 120
300 CALL JOYST(1,X,Y)
310 Y=Y/4+2
320 ON Y GOTO 330,160,360
330 IF FRE(FREQ)=14000 THEN 160
340 FRE(FREQ)=FRE(FREQ)+1
350 GOTO 160
360 IF FRE(FREQ)=110 THEN 160
370 FRE(FREQ)=FRE(FREQ)-1
380 GOTO 160
390 REM
400 REM RAUSCH-GENERATOR
410 REM
420 FREQ=0
430 FRE(FREQ)=-1
440 GOSUB 990
450 CALL KEY(1,K,S)
460 IF S<>0 THEN 590
470 CALL JOYST(1,X,Y)
480 Y=Y/4+2
490 ON Y GOTO 500,440,530
500 IF FRE(FREQ)=-8 THEN 440
510 FRE(FREQ)=FRE(FREQ)-1
520 GOTO 440
530 IF FRE(FREQ)=-1 THEN 440
540 FRE(FREQ)=FRE(FREQ)+1
550 GOTO 440
560 REM
570 REM END-CHECK
580 REM
590 CALL CLEAR
600 PRINT "GENERATOR 1 =":TAB(10);
FRE(1);"Hz": :
610 PRINT "GENERATOR 2 =":TAB(10);
FRE(2);"Hz": :
620 PRINT "GENERATOR 3 =":TAB(10);
FRE(3);"Hz": :
630 PRINT "RAUSCH-GENERATOR = ";
FRE(0)
640 PRINT : : :"AENDERUNG = J":
"STOP = S"
650 GOSUB 990
660 CALL KEY(3,K,S)
670 IF S=0 THEN 650
680 IF K=74 THEN 740
690 IF K<>83 THEN 650
700 STOP
710 REM
720 REM AENDERUNG
730 REM
740 CALL CLEAR
750 PRINT "WELCHE FREQUENZ AENDERN ?
    (0 = RAUSCHEN)":"(1 = FREQ 1)":
    "(2 = FREQ 2)":"(3 =FREQ 3)"
760 GOSUB 990
770 CALL KEY(3,K,S)
780 IF S=0 THEN 760
790 IF K=48 THEN 420
800 IF (K<49)+(K>51)THEN 760
810 FRA=K-48
820 CALL CLEAR
830 PRINT "AENDERUNG FREQUENZ";FRA
840 GOSUB 990
850 CALL KEY(1,K,S)
860 IF S<>0 THEN 590
870 CALL JOYST(1,X,Y)
880 Y=Y/4+2
890 ON Y GOTO 900,840,930
900 IF FRE(FRA)=14000 THEN 840
910 FRE(FRA)=FRE(FRA)+1
920 GOTO 840
930 IF FRE(FRA)=110 THEN 840
940 FRE(FRA)=FRE(FRA)-1
950 GOTO 840
960 REM
970 REM SUB-ROUTINE TONERZEUGUNG
980 REM
990 ON FREQ+1 GOTO 1000,1020,
1040,1060
1000 CALL SOUND(-4000,FRE(1),0,
FRE(2),0,FRE(3),
    0,FRE(0),0)
1010 RETURN
1020 CALL SOUND(-4000,FRE(1),0)
1030 RETURN
1040 CALL SOUND(-4000,FRE(1),0,
FRE(2),0)
1050 RETURN
1060 CALL SOUND(-4000,FRE(1),0,
FRE(2),0,FRE(3),0)
1070 RETURN
```

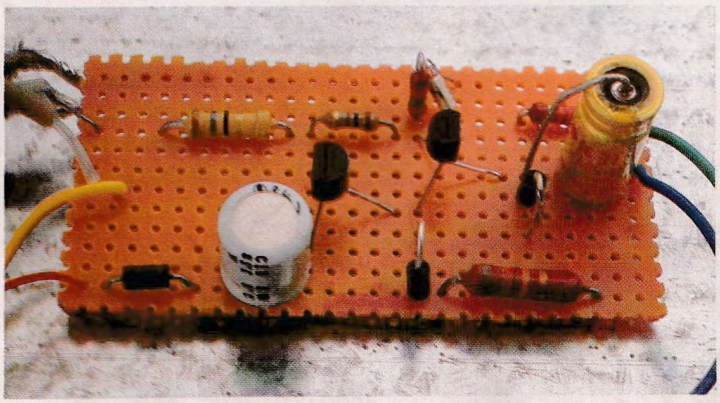
Scharfe Sache — scharfes Bild

Wußtet ihr schon, daß die Sterne
bei 'Parsec' in den
verschiedensten Farben funkeln?
Mit unserem BNC-Adapter wird's richtig bunt
auf dem Bildschirm!





Mit wenig Mühe und Kosten holt ihr ein besseres Bild aus dem PHA-2036.



Die Abgriffe von rechts nach links: Video, Masse und +12 Volt (rot) mit Audio (gelbes Kabel).

links: Aufbau im Freistil

Im Heft 5/84 von COMPUTER Praxis las ich den Artikel "Monitor statt Flimmerkiste", in dem beschrieben wird, wie man den Atari über einen BNC-Stecker an einem entsprechenden Monitor anschließt.

Als TI-User fühlte ich mich sofort herausgefordert. Ich drehte meinen 99/4A um und betrachtete die Steckdose, an der der HF-Modulator angschlossen wird. Da muß doch wohl irgendwas zu machen sein!

Schnell war ein entsprechender Sechs-Pol DIN-, ein BNC- und ein Chinch-Stecker gekauft. Ebensoschnell war mir klar; so einfach geht das nicht.

Also forderte ich von TI Freising die Schaltpläne für den TI 99/4A an. Aus den Plänen geht hervor, daß an der Steckdose des Rechners folgende Signale anliegen:

—PIN 1 = +12 Volt
—PIN 2 = Y-Signal
—PIN 3 = (R-Y)-Signal
—PIN 4 = (B-Y)-Signal
—PIN 5 = Audio-Signal
—PIN 6 = Masse

Das Monochrom-Bild ist also kein Problem; man nehme Y (PIN 2) und Masse (PIN 6) sowie eine weitere Leitung für den guten Ton (PIN 5).

Ich wollte aber ein Farb-Bild. Und das bekommt man nur bei entsprechender Umwandlung. Mein Gedankengang war, daß ja wohl irgendwo im HF-Modulatorteil ein Video-Signal, wie es für den BNC-Anschluß benötigt wird, vorliegen muß, bevor es auf die Hochfrequenz aufmoduliert wird.

Die Oszi-Fahndung war von Erfolg gekrönt: an einem Beinchen des im Bild rot eingekreisten Transistors (Emitter) liegt ein stabiles Video-Signal. Schnell war mein vorbereitetes Kabel an dieses

Beinchen angeklemmt — aber Totalpleite! Alles was ich zu sehen bekam war ein schwarzer Bildschirm. Ganz offensichtlich war das Signal ganz frech zusammengebrochen.

Stückliste

T1	= BC 548 B
T2	= BC 415 C
D1—D3	= 1 N 4001
C1	= 47 µF
C2	= 100 µF
C3	= 4,7 pF
R1	= 8,2 kΩ
R2	= 12 kΩ
R3	= 180 Ω
R4	= 100 Ω
R5	= 18 Ω

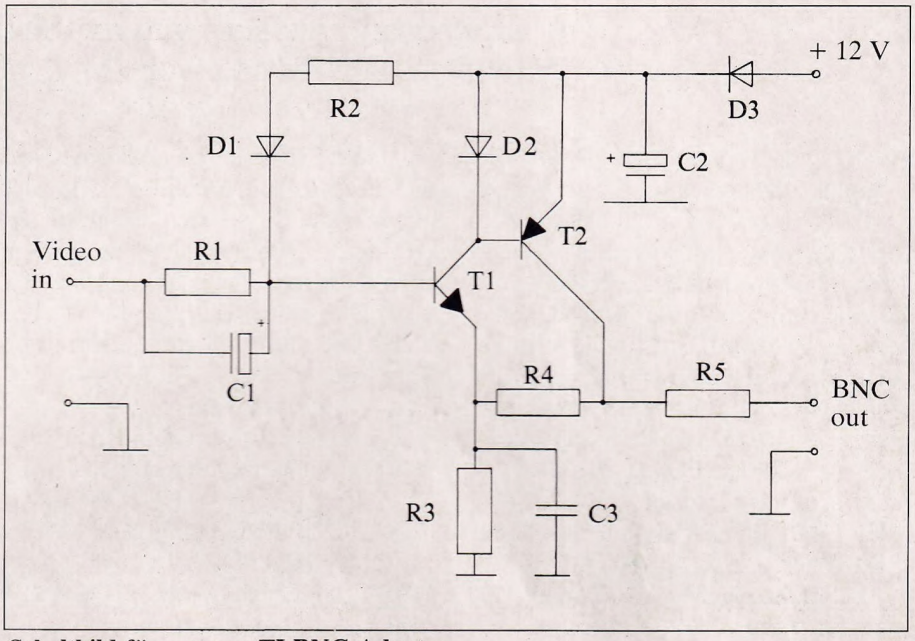
Ein Verstärker war also fällig. Als Computer- und somit IC-Fan suchte ich ein solches Verstärker-IC, konnte es aber nirgends erwerben.

Mich packte die Wut und ich den Lötkolben wieder aus. Nach intensivem Studium verschiedener Fachbücher faßte ich den Entschluß, es mit einem zweistufigen Verstärker zu wagen.

So wird's gemacht!

Das Video-Signal, das an dem bewußten Transistor abgegriffen wird, läuft über T1 und T2 zum BNC-Stecker. T1 dient dabei der Impedanz-Anpassung und ist die erste Verstärker-Stufe. Den Rest besorgt uns Transistor T2, der den benötigten Ausgangspegel von etwa 0,7 Volt liefert. Der Stromverbrauch liegt bei ca. 35 bis 50 mA, je nach Belastung, d.h. die Versorgung kann direkt vom Modulator abgezapft werden.

Masse holen wir uns vom Gehäuse. Und damit haben wir das erhoffte Video-Signal, das auf den BNC-Eingang eines Monitors oder eines entsprechend ausgerüsteten Farb-Fernsehers (PAL) gelegt wird. Was noch fehlt, ist der Ton, das Audio-Signal. Auch das finden wir im Modulator. Mit einem abgeschirmten Kabel und einem Chinch-Stecker wird es an der entsprechenden Steckdose des Monitors angestöpselt. Aber aufpassen! Die Masse darf am Chinch-Stecker nicht angelötet werden, da wir sonst eine Masse-Schleife legen.



Schaltbild für unseren TI-BNC-Adapter

So habe ich mir von meinem TI 99/4A ein Bomben-Bild auf den Fernseher geholt. Wo genau die einzelnen Anschlüsse im Modulator gemacht werden müssen, könnt ihr im Bild sehen.

Die Kosten der Schaltung liegen bei ca. 20 bis 25 Mark, je nach örtlichen Möglichkeiten. ■

Rüdiger Bauszus

U N S E R
ANGEBOT FÜR
SIE

Arcade
Professional
Super
Joystick
128,-

...über 1000
Original-
Programme

Atari TI IBM
Commodore Sinclair

Fordern Sie
unseren
informativen
Katalog an!
System angeben!!!

10 (zehn)
Daten-
Cassetten '10
Nur
15,- DM

Wenn Preis,
Qualität
und
Service
stimmen
sollen!
EASY—Soft
Olaf Bauer
Kritenbarg 44
2 Hamburg 65
Tel. (040) 606 24 87
Auch nachts

MOONRAKER

Beat Schönauer besucht unseren quittengelben Trabanten in X-BASIC von der Schweiz aus.

Wir wollen euch diesmal ein Listing eines Lesers vorstellen. Als Vorbild hat das wohl vielen bekannte "Moon-Patrol" gedient. Mit viel Geschick ist das Spiel an den TI adaptiert worden. Ihr braucht zum Spielen das Extended BASIC-Modul. Wer keine RAM-Erweiterung hat, muß drei Befehle aus dem Listing herausnehmen:
Zeile 460: CALL INIT::
CALL LOAD(-31878,1)
Zeile 470: ganz löschen
Dann läuft das Programm auch ohne RAM-Pack.

Die Zeilen ab 2000, 3000 und 4000 sind für eigene Ideen vorgesehen. Wir würden uns sehr freuen, in einer der nächsten Ausgaben einige eurer Vorschläge abdrucken zu können.

Wer sich von dem Schriftzug "MOONRAKER" mitten auf dem Bildschirm genervt fühlt, der lösche einfach die Zeile 820, schon ist er erlöst.

Das Listing und die Dokumentation sind uns von Beat Schönauer aus der Schweiz zugeschickt worden.

Spielbeschreibung:

Ziel des Spiels ist das Überspringen von Hindernissen. Pro gefahrenem Meter gibt es 10 Punkte. Bei Erreichen von 500 fehlerfrei gefahrenen Metern wird eine neue Runde begonnen. Bei jeder Runde gibt es, je nach Geschwindigkeitsfaktor, Bonuspunkte. Das Spiel ist nicht schwierig, und es läßt sich erweitern. Dazu sind die Programmzeilen 2000, 3000 und 4000 vorgesehen.

Übersicht:

100— 520	Titelroutine
530— 660	Charakterdefinitionen
670	Start Moonraker Programm
720— 770	Initialisierung Color + Char
780— 980	Erstellung des Bildes
960	Start Moonraker
990	Start Hügel und Krater
1040—1230	Spielschleife
1180	springt durch Zufall nach: 2000—kein Hindernis 3000—Hindernis 1—4 4000—Raketen 1—4
1240—1360	Programm-Ende; Frage, ob neues Spiel
1370—1440	Sub neuer Highscore wurde erspielt
1450—1490	neue Spielrunde
1500—1540	neue Spielgeschwindigkeit setzen
1550—1670	Sub Sprung Moonraker
1680—1740	Sub trifft Hindernis
2000	Sub keine zusätzlichen Hindernisse
3000	Sub vorgesehen für Hindernisse 1—4
4000	Sub vorgesehen für Raketen 1—4

Variablen:

PUNKTE	Spielpunktezähler
HSCORE	Highscore Punktezähler
MOON	Anzahl Moonraker
V	Geschwindigkeitsfaktor
RAD	Bewegung Räder und Rundenzähler
X1, X2, Y1, Y2	Position Sub Sprung des Moonrakers
I	Schleifenzähler
K, S	CALL KEY-Abfrage
X, Y	CALL JOYST-Abfrage
C1, C2	CALL COINC-Abfrage

Sprites:

5	Moonraker
8	Räder des Moonrakers
6+7	Hügel und Krater

Spielerweiterung:

Zeile 660 springt durch Zufall nach:
50%—5000 kein Hindernis
30%—5100 Hindernis1—4
20%—5500 Raketen 1—4

Durch Sprites bringt ihr sie ins Spiel.

CHAR 100—103 Rakete 1

1 1E 78	...XXXX..XXXX...
2 0E 70	....XXX..XXX....
3 06 60	.....XX..XX.....
4 02 40	......X..X......
5 02 40	......X..X......
6 02 40	......X..X......
7 03 C0	......XXXX......
8 02 40	......X..X......
9 02 40	......X..X......
10 03 C0	......XXXX......
11 02 40	......X..X......
12 02 40	......X..X......
13 03 C0	......XXXX......
14 01 80	.......XX.......
15 01 80	.......XX.......
16 01 80	.......XX.......

CHAR 104—107 Rakete 2

1 00 00	....
2 1D 70	...XXX.X.XXX....
3 0D 60	....XX.X.XX.....
4 05 40	.....X.X.X......
5 07 C0	.....XXXXX......
6 07 C0	.....XXXXX......
7 06 C0	.....XX.XX......
8 06 C0	.....XX.XX......
9 06 C0	.....XX.XX......
10 06 C0	.....XX.XX......
11 06 C0	.....XX.XX......
12 07 C0	.....XXXXX......
13 07 C0	.....XXXXX......
14 07 C0	.....XXXXX......
15 03 80	......XXX.......
16 01 00	.......X........

CHAR 108—111 Rakete 3

1 01 80	.......XX.......
2 01 80	.......XX.......
3 1F F8	...XXXXXXXXX...
4 1F F8	...XXXXXXXXX...
5 17 E8	...X.XXXXXX.X...
6 1C 38	...XX......XX...
7 1C 38	...XX......XX...
8 17 E8	...X.XXXXXX.X...
9 1C 38	...XX......XX...
10 1C 38	...XX......XX...
11 0F F0	....XXXXXXXX....
12 07 E0	.....XXXXXX.....
13 03 C0	......XXXX......
14 01 80	.......XX.......
15 01 80	.......XX.......
16 01 80	.......XX.......

CHAR 112—115 Rakete 4

```
 1 E6 67    XXX..XX..XX..XXX
 2 66 66    .XX..XX..XX..XX.
 3 3F FC    ..XXXXXXXXXXXX..
 4 13 C8    ...X..XXXX..X...
 5 12 48    ...X..X..X..X...
 6 1E 78    ...XXXX..XXXX...
 7 12 48    ...X..X..X..X...
 8 1E 78    ...XXXX..XXXX...
 9 12 48    ...X..X..X..X...
10 13 C8    ...X..XXXX..X...
11 1F F8    ...XXXXXXXXXX...
12 03 C0    ......XXXX......
13 01 80    .......XX.......
14 01 80    .......XX.......
15 01 80    .......XX.......
16 01 80    .......XX.......
```

CHAR 120—123 Hügel

```
 1 00 00    ................
 2 00 00    ................
 3 00 00    ................
 4 00 00    ................
 5 00 00    ................
 6 05 88    .....X.XX...X...
 7 07 E8    .....XXXXXX.X...
 8 07 E8    .....XXXXXX.X...
 9 07 EA    .....XXXXX.X.X.
10 0F FE    ....XXXXXXXXXX.
11 1F FF    ...XXXXXXXXXXXX
12 1F FF    ...XXXXXXXXXXXX
13 3F FF    ..XXXXXXXXXXXXX
14 7F FF    .XXXXXXXXXXXXXX
15 FF FF    XXXXXXXXXXXXXXXX
16 FF FF    XXXXXXXXXXXXXXXX
```

CHAR 124—127 Krater

```
 1 FF FF    XXXXXXXXXXXXXXXX
 2 FF FF    XXXXXXXXXXXXXXXX
 3 FF FF    XXXXXXXXXXXXXXXX
 4 FF FF    XXXXXXXXXXXXXXXX
 5 FF FD    XXXXXXXXXXXXXX.X
 6 BF FD    X.XXXXXXXXXXXX.X
 7 1F F8    ...XXXXXXXXXX...
 8 1F F8    ...XXXXXXXXXX...
 9 1F F8    ...XXXXXXXXXX...
10 1F E0    ...XXXXXXXX.....
11 1F C0    ...XXXXXXX......
12 06 00    .....XX.........
13 06 00    .....XX.........
14 00 00    ................
15 00 00    ................
16 00 00    ................
```

```
100 !
110 RANDOMIZE
120 !
130 ! MOONRAKER
140 ! BY BEAT SCHOENAUER
150 ! ALPENSTRASSE 32
160 ! CH-2540 GRENCHEN
170 !{9*(SPACE)}{8*(=)}
180 ! SEPTEMBER 1983
190 ! PROGRAMMBEGINN
200 !
210 !
220 !TITELSCREEN STEINWERFER
230 M$="MOONRAKER BY BEAT SCHOENAUER"
240 Z=10 !ZEICHEN ZEILE
250 CALL CLEAR :: CALL SCREEN(2):: FOR N=0 TO 14 ::
     CALL COLOR(N,5,1):: NEXT N
260 CALL CHAR(139,""):: CALL CHAR(140,"{8*(0)}
    1"):: CALL CHAR(141,"{6*(0)}
    281C3808"):: CALL CHAR(142,"00025438345804A0")
270 CALL SPRITE(#1,139,16,1,1)
280 FOR N=1 TO 28
290 C=ASC(SEG$(M$,N,1)):: IF C>32 THEN 310
300 FOR D=1 TO 60 :: NEXT D :: GOTO 350
310 CALL LOCATE(#1,Z*8-7,N*8+9)
320 CALL MAGNIFY(1):: FOR Q=0 TO 2 ::
     CALL PATTERN(#1,Q+140):: FOR D=1 TO 15 ::
     NEXT D :: NEXT Q
330 CALL LOCATE(#1,Z*8-13,N*8+4):: CALL MAGNIFY(2)::
     FOR Q=0 TO 1 :: CALL PATTERN(#1,Q+140)::
     FOR D=1 TO 4 :: NEXT D :: NEXT Q
340 CALL PATTERN(#1,142):: CALL HCHAR(Z,N+2,C)::
     CALL PATTERN(#1,139)
350 NEXT N
360 ! COSMIC ADVENTURES
370 M$="COSMIC ADVENTURES" ! HIER KANN EIGENER TITEL
     EINGEFUEGT WERDEN
380 AUT$="B SCHOENAUER" ! AUTOR MAX 13 BUCHSTABEN
390 CALL CHAR(127,"3C4299A1A199423C"):: 
    CALL CHAR(128,"000101{45*(0)}")
400 CALL CHAR(132,"{7*(0)}
    101C363371F0C0810304884038080800C0C0E3E6ECF830100
    80C1221C0")
410 CALL CHAR(136,"03070F0F0F0F0F0F0707070703030301 01C
    0E0F0F0F0F0E0E0C0C0C080808")
420 FOR N=5 TO 12 :: CALL COLOR(N,14,1):: NEXT N ::
     DISPLAY AT(15,16):CHR$(127)&" Programm by" ::
     DISPLAY AT(17,16):AUT$
430 DISPLAY AT(7,12):M$ :: DISPLAY AT(24,2):
    "Beginn mit beliebiger Taste"
440 CALL MAGNIFY(3):: CALL
     SPRITE(#1,136,1,134,81,0,0)
450 FOR N=2 TO 26 :: CALL SPRITE(#N,128,1,96,86,
    RND*30-15,RND*30-15):: NEXT N
460 CALL INIT :: CALL LOAD(-31878,1)::
     FOR N=2 TO 26 :: CALL LOCATE(#N,96,86)::
     CALL COLOR(#N,INT(N/2)+2):: NEXT N
470 CALL LOAD(-31878,26)
480 CALL COLOR(13,5,1):: DISPLAY AT(16,9):
    CHR$(132);CHR$(134);
490 DISPLAY AT(17,9):CHR$(133);CHR$(135);
500 CALL COLOR(#1,12):: CALL COLOR(#1,2)::
     FOR N=2 TO 26 :: CALL LOCATE(#N,96,86)::
     CALL KEY(3,K,R):: IF R<>0 THEN 520
510 NEXT N :: GOTO 500
520 CALL COLOR(#1,12)
530 !
540 !
550 ! CHAR 96-103
560 DATA ,,,{16*(F)},1E0E060202030302,
    0203020203010101,7870604040C0C040,
    40C04040C0808080
570 ! CHAR 104-111
580 DATA 0005050507070606,0606060707070301,
    7870604040 40C0C040,40C04040C0808080,
    01011F1F171C1C17,1C1C0F0703010101,
    8080F8F8E83838E8,3838F0E0C0808080
590 ! CHAR 112-119
```

```
600 DATA E6663FF13121E1212,1E12131F03010101,
    6766FCC848784848,7848C8F8C0808080,{8*(0)}
    30303CFF,FF80F0FF991818,8080808080808E0,
    F87C0EFFB1300
610 ! CHAR 120-127
620 DATA {11*(0)}50707,070F1F1F3F7FFFFF,{9*(0)}
    888E8E8,EAFE{12*(F)},{10*(F)}
    BF1F1F,1F1F1F0606,{9*(F)}DFDF8F8,F8E0C
630 ! CHAR 128-135
640 DATA 00183C66663C18,,183C66663C18,,183C66663C18,
    ,00183C66663C18,
650 ! CHAR 136-143
660 DATA 2020F82022020F02,925438FF38549211,
    02070240E04,2070240E0440E040,894A2CF81F2C4A89,
    202020FF2020,00044424140C7F04,101038FE381010
670 !
680 !  ANFANG
690 !
700 CALL CLEAR
710 DISPLAY AT(10,6):"MOONRAKER":
720 !
730 ! INITIALISIERUNG
740 !
750 FOR I=96 TO 143 :: READ A$ :: CALL CHAR(I,A$)::
     NEXT I
760 CALL COLOR(1,16,1,2,16,1,3,16,1,4,16,1,5,16,1,6,
    16,1,7,16,1,8,16,1,9,11,1,14,16,1)
770 MOON=5 :: V=0 :: ZA=1 :: PUNKTE=0
780 !
790 ! BILD ERSTELLUNG
800 !
810 CALL CLEAR :: CALL DELSPRITE(ALL)
820 DISPLAY AT(16,3):"MOONRAKER":
830 CALL MAGNIFY(3)
840 CALL SCREEN(5)
850 CALL HCHAR(20,1,99,128)
860 FOR I=1 TO 20
870 X=INT(RND*11+3):: Y=INT(RND*32)+1
880 CALL HCHAR(X,Y,136+INT(RND*9))
890 NEXT I
900 DISPLAY AT(1,1):"SCORE";
910 DISPLAY AT(1,10):"HIGHSCORE";
920 DISPLAY AT(1,20):"RUNDE";
930 DISPLAY AT(1,26):V+1;
940 DISPLAY AT(2,26):MOON;
950 DISPLAY AT(2,10):HSCORE;
960 ! START AUTO
970 !
980 CALL SPRITE(#5,116,10,140,10,0,5,#8,132,2,153,
    10,0,5)
990 !
1000 ! START HINDERNISSE
1010 !
1020 CALL SPRITE(#6,124,5,150,100,#7,120,11,140,220)
1030 GOSUB 1500
1040 !
1050 !
1060 ! START SPIELSSCHLEIFE
1070 !
1080 !
1090 CALL KEY(1,K,S)
1100 CALL JOYST(1,Y,X)
1110 IF S<>0 THEN GOSUB 1550
1120 CALL MOTION(#5,0,Y*4,#8,0,Y*4)
1130 CALL COINC(#5,#7,15,C2)! HINDERNISSE
1140 CALL COINC(#5,#6,15,C1)! HINDERNISSE
1150 IF C1+C2<>0 THEN GOTO 1680
1160 IF RAD/2=INT(RAD/2)THEN CALL PATTERN(#8,128)
    ELSE CALL PATTERN(#8,132)
1170 ZUFALL=INT(RND*10+1)
1180 ON ZUFALL GOSUB 2000,3000,2000,4000,3000,2000,
    3000,2000,4000,2000
1190 !
1200 RAD=RAD+1
1210 IF RAD=500 THEN 1450
1220 DISPLAY AT(2,1):PUNKTE;
1230 GOTO 1090 !GOTO START SPIELSCHLEIFE
1240 ! PROGRAMMENDE
1250 IF PUNKTE>HSCORE THEN GOSUB 1370
1260 DISPLAY AT(12,8):"<J> NEUES SPIEL"
1270 DISPLAY AT(2,10):HSCORE;
1280 DISPLAY AT(14,8):"<N> PGM ENDE"
1290 DISPLAY AT(9,8):"< GAME OVER >"
1300 CALL KEY(3,K,S)
1310 FOR VER=1 TO 10 :: NEXT VER
1320 DISPLAY AT(9,8):"<{11*(SPACE)}>"
1330 IF S=0 THEN 1290
1340 IF K=74 THEN 770 ! BILD ERSTELLUNG
1350 IF K<>78 THEN 1300
1360 END
1370 FOR I=1 TO 5 :: CALL SOUND(330,550,0,660,0,770,
    0)
1380 DISPLAY AT(1,10):"{9*(SPACE)}";
1390 FOR VER=1 TO 30 :: NEXT VER
1400 DISPLAY AT(1,10):"HIHGSCORE";
1410 FOR VER=1 TO 30 :: NEXT VER :: NEXT I
1420 DISPLAY AT(2,10):HSCORE;
1430 HSCORE=PUNKTE
1440 RETURN
1450 ! NEUE RUNDE
1460 PUNKTE=PUNKTE+V*100
1470 MOON=MOON+1
1480 RAD=0
1490 GOTO 960 ! START AUTO
1500 REM RUNDMOT
1510 V=V+1 :: IF V>15 THEN V=5 !BESCHLEUNIGUNG DER
     FAHRZEUGE
1520 CALL SOUND(390,780,5,790,6,770,7,-1,0)
1530 CALL MOTION(#6,0,-(V+8),#7,0,-(V+8))
1540 RETURN
1550 REM SUB SPRUNG
1560 CALL POSITION(#5,X1,Y1,#8,X2,Y2)
1570 CALL MOTION(#5,-25,35,#8,-25,35)
1580 FOR VER=1 TO 90 :: NEXT VER
1590 CALL MOTION(#5,0,0,#8,0,0)
1600 CALL MOTION(#5,25,0,#8,25,0)
1610 FOR VER=1 TO 90 :: NEXT VER
1620 CALL MOTION(#5,0,0,#8,0,0)
1630 IF Y1>210 THEN Y1=Y1-245
1640 IF Y2>210 THEN Y1=Y2-245
1650 CALL LOCATE(#5,X1,Y1+40,#8,X2,Y1+40)
1660 PUNKTE=PUNKTE+39
1670 RETURN
1680 REM SUB TREFFER
1690 CALL DELSPRITE(#5,#8)
1700 FOR VER=1 TO 5 :: CALL
      SOUND(50,770,7,660,6,550,5)::
      CALL SOUND(70,550,5,440,4,330,3)::
      CALL SOUND(100,220,2,330,3,110,0):: NEXT VER
1710 MOON=MOON-1
1720 IF MOON<1 THEN 1240
1730 PUNKTE=PUNKTE-100 :: V=V-1
1740 GOTO 930
2000 ! KEIN HINDERNIS
2010 PUNKTE=PUNKTE+10
2020 RETURN
3000 ! HINDERNIS 1-4
3010 RETURN
4000 ! RAKETEN 1-4
4010 RETURN ■
```

Beat Schönauer

SOFTWARE IM TEST

Cerberus

Uns liegt hier ein Programm auf Diskette vor, das auch in den TI 99 Special-Büchern als Assembler-Listing abgedruckt ist.

Leider kann es nur mit Laufwerk, Speichererweiterung und XBASIC- bzw. Editor/Assembler-Modul geladen und gespielt werden.

Hat man das recht umfangreiche Programm aber erst einmal im Speicher, wird man voll entschädigt: Der Spieler wird von einer wohlklingenden Melodie empfangen — Johann Sebastian Bach läßt schön grüßen.

Das Spiel beweist dann, daß Graphik mit dem TI nicht zwangsläufig langsam sein muß. Mit seinem Schiff steuert der Spieler durch ein Höhlen-Labyrinth in die Hölle, wo Cerberus der Höllenhund, seinen Schatz bewacht. Ihn gilt es selbstverständlich zu erobern.

Doch auch die Fahrt zur Hölle verläuft nicht ohne Komplikationen, denn die Helfer des Cerberus, Batmoy, das Fledermausmonster, Spiderball und Suakey, greifen erbarmungslos an.

Gesteuert wird das Schiff wahlweise mit Joystick oder Tastatur.

Graphik:

Sound:

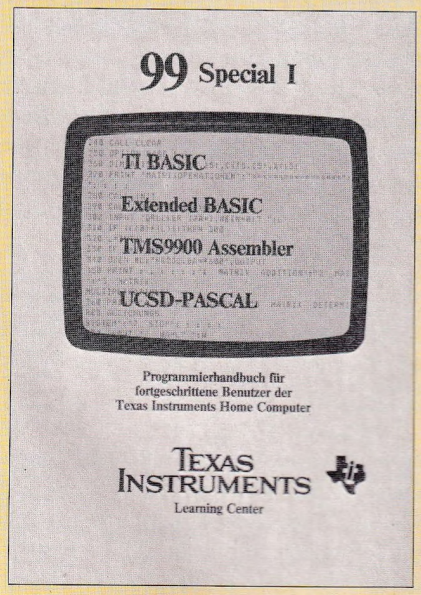
Spielidee:

Testergebnis:

Mehrsprachig

Aus dem TI Learning Center liegt uns ein weiteres Buch zum TI 99/4A vor, das "99 Special I". Hierbei handelt es sich um wesentlich mehr als eine Ansammlung von Listings, nämlich ein Programmierhandbuch für Benutzer des leider nicht mehr produzierten Rechners.

Neben dem Hauptteil der Beispiele, der in TI-BASIC oder TI-Extended BASIC erstellt wurde, tauchen solche in TMS 9000 Assembler und UCSD-Pascal auf. Dieses Buch bietet jedoch keine Grundkenntnisse der Sprachen. Es wird vorausgesetzt, daß man zumindest von ihnen gehört hat.



99 Special I
Alma und Johann Peschetz
TI Learning Center 295/73830
ISBN 3-88078-043-9
298 Seiten 49,50 DM

Um parallel zum Buch mitarbeiten zu können, sollte man wenigstens über den TI 99/4A mit Erweiterungsbox, Disk-Controller, ein Laufwerk sowie die Module XBASIC und/oder MINI MEMORY verfügen.

Thematisch befassen sich die Programme mit recht viel Mathematik, Graphik und Musik. Alle Listings sind ausgezeichnet dokumentiert, aus allen kann man sich Teil- oder Gesamtlösungen für eigene Problemstellungen rauspicken. Es gibt genaue Erklärungen, was wo und wie gemacht wird.

Der Schwierigkeitsgrad steigert sich nach und nach; Lernerfolg scheint vorprogrammiert, wenn der Leser systematisch mitarbeitet.

Keine Rose ohne Dornen: Es handelt sich auch hierbei wieder einmal um ein Machwerk aus dem Matrixdrucker, natürlich auch noch verkleinert. Das mag zwar unheimlich computerig aussehen und außerdem die Kosten für den "Saezzer" sparen, aber das Zeug läßt sich nun mal nicht so gut lesen wie gesetzte Schrift.

Prinzipiell ist dieses Buch durchaus zu empfehlen, besonders wenn ihr über das BASIC-Beginner-Stadium herauswachsen wollt.

ThoK

ACHTUNG!
TI-99 / 4A Besitzer
ab sofort wieder lieferbar:

	DM
Extended Basic-Modul	290,–
Mini Memory	290,–
Editor Assembler	198,–
Parsec	80,–
und alle Atarisoft-Module für TI-99	
Thermodrucker PC100C	298,–

Alle Preise incl. MwSt. zuzügl. 5,– DM Versandkosten. Lieferung per Nachnahme oder Vorkasse, ab 200,– DM Versandkostenfrei.

PSR Programm-Service REIS GmbH
D-5584 Bullay, Bergstr. 80, ☎ 06542/2715

Parallel und seriell

3. Folge
Wie und warum Schnittstellen funktionieren sollen, haben wir schon gesehen. Wenn's aber doch nicht läuft, nützt das blanke Wissen um Normen so gut wie nichts.

Eine serielle Schnittstelle kann Daten Bit für Bit ausgeben und/oder einlesen. Wie sie das macht, steht in den Sternen (oder, wenn man Glück hat, im Handbuch). Erst durch die V.24 (oder RS-232-C)-Norm werden wichtige technische Daten festgelegt. Dies gilt besonders für die verwendeten Spannungen. Deshalb wiederholen wir noch einmal die Warnung aus der vorigen Folge: schließt niemals einen Computer mit einer einfachen seriellen Schnittstelle direkt an ein Gerät mit V.24 Schnittstelle an — es würde ihm nicht bekommen.

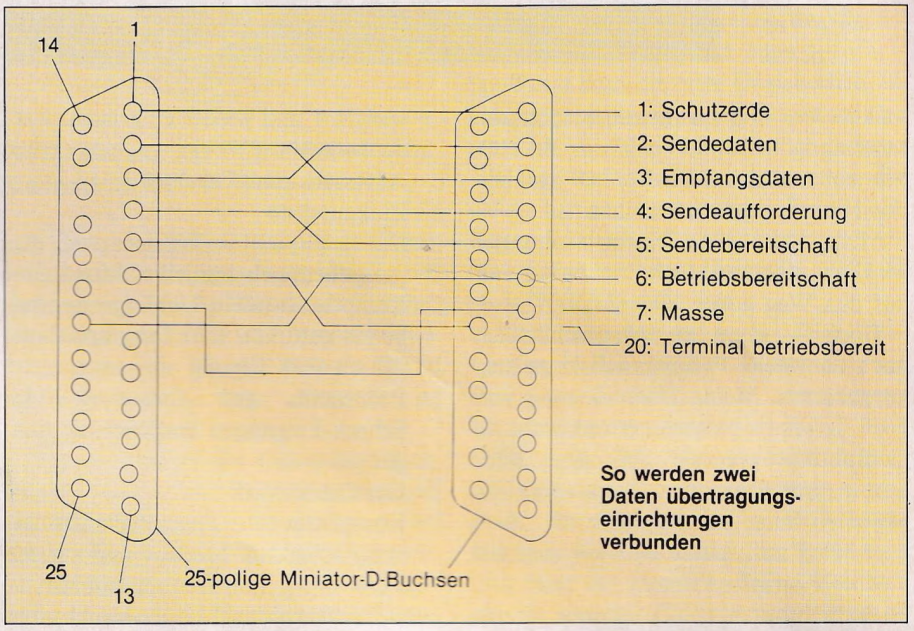
Das Verbindungskabel — trotz Normung ein Problem

Sehen wir uns einmal die Datenleitungen an. Die gesendeten Daten liegen an Anschluß 2 (Ausgang), empfangene Daten liegen an Anschluß 3 (Eingang) an. Wenn wir zwei Computer miteinander verbinden wollen, wird der eine Computer Daten an Anschluß 2 ausgeben — und der andere Computer muß sie an Anschluß 3 empfangen! Deshalb müssen wir die Sende- und Empfangsleitungen unterwegs überkreuzen. Das gleiche gilt für die anderen Leitungspaare — 20 und 6, sowie 4 und 5.

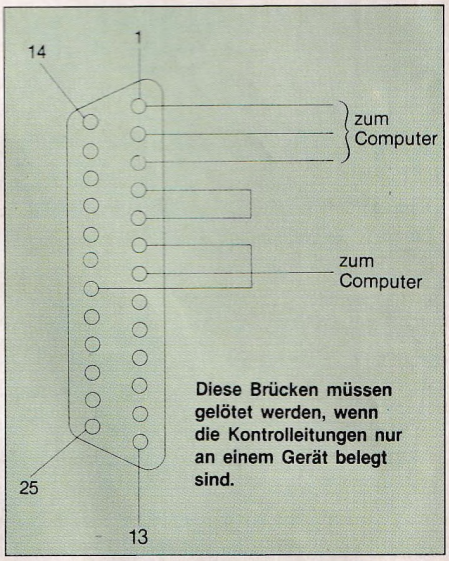
Dieses nötige Überkreuzen von Leitungen ist immer wieder ein Problem. Stellt euch eine andere mögliche Verbindung vor — von einem Computer zu einem Drucker zum Beispiel. Dann ist klar, daß der Computer der 'Chef' ist, der Drucker nur williger 'Diener'. Genau für solche Fälle hat man eine Unterscheidung getroffen. Zwischen 'Datenübertragungseinrichtungen' — das sind zum Beispiel Computer oder Terminals — und 'Datenendeinrichtungen', zum Beispiel Druckern oder Modems. Um nun das Überkreuzen der Leitungen zu sparen, sind bei den Peripheriegeräten ('Datenendeinrichtungen') Ein- und Ausgänge vertauscht. Die Bezeichnungen der Anschlüsse bleiben gleich. Ein Beispiel: der Anschluß 2 heißt immer 'Sendedaten'. Bei einem Computer ist dieser Anschluß ein Ausgang (Daten werden gesendet). Bei einem Peripheriegerät ist an diesem Anschluß aber ein Eingang (gesendete Daten werden hier empfangen). Durch diesen 'Kunstgriff' kann man die Geräte direkt, also ohne Kabel zu überkreuzen, verbinden.

Schön und gut. In der Praxis taucht allerdings häufiger Verwirrung auf, weil man nicht weiß, wie die Anschlüsse an den Geräten sind. Ein Anhaltspunkt: hat das Gerät einen Stecker als Anschluß, so sollte es als Datenübertragungseinrichtung arbeiten, hat es eine Buchse, sollte es ein Peripheriegerät sein. Ein anderer Anhaltspunkt: Computer und Terminals sind meist als Datenübertragungseinrichtungen beschaltet, Modems und Drucker meist als Dateneinrichtung. So sollte es sein, aber man erlebt man oft Wunder.

Oft haben Heimcomputer — wenn überhaupt — nur eine Sparversion der V.24 Schnittstelle. Dort sind außer den



So werden zwei Daten übertragungseinrichtungen verbunden



Diese Brücken müssen gelötet werden, wenn die Kontrolleitungen nur an einem Gerät belegt sind.

Masseleitungen nur die Sende- und Empfangsleitungen vorhanden. Wenn man nun ein Gerät anschließen will, das auch die Kontrolleitung an den Anschlüssen 4, 5, 6 und 20 benutzt, muß man dafür sorgen, daß an den freien Eingängen vernünftige Signale anliegen. Hier hilft ein einfacher Trick: wir bauen in den Stecker, der an dieses Gerät angeschlossen wird, einige Brücken ein und verwandeln so die 'Luxus'-Schnittstelle in eine Sparversion, die mit unserem Computer verbunden werden kann.

Es läuft — aber nicht richtig

Wenn das richtige Kabel endlich fertig ist, gibt es oft weitere Probleme. Zwar erscheinen Zeichen auf dem Bildschirm oder Drucker, aber die haben nicht viel mit dem zu tun, was übertragen wurde.

Erste Frage: Ist die Übertragungsgeschwindigkeit (Baudrate) richtig eingestellt?

Zweite Frage: stimmt das Datenformat? In der letzten Folge waren wir davon ausgegangen, daß jedes Byte als eine Folge von 10 Bit übertragen wird: Startbit, 8 Datenbits, Stopbit. Da aber für ASCII-Zeichen (siehe Folge 1) nur 7 Bits benötigt werden, werden oft auch nur 7 Datenbits übertragen. Und wenn es ganz penibel zugeht, wird zusätzlich zu den Datenbits noch ein Paritätsbit gesendet. Dieses Bit bildet eine Quersumme über die Datenbits und ermöglicht so eine einfache Prüfung auf Übertragungsfehler. Diese unterschiedlichen Einstellungen bezeichnet man als Datenformate. In unserer Tabelle sind die gebräuchlichsten zusammengestellt.

Was ist nun zu tun, wenn die Datenformate der miteinander verbundenen Geräte nicht übereinstimmen? An einem Gerät muß das Format so geändert werden, daß es mit dem anderen übereinstimmt. Am einfachsten ist das zu machen, wenn ein Gerät über Schalter zur Einstellung des Datenformats verfügt. Dies ist besonders bei Druckern und Terminals oft der Fall, auch wenn die Schalter manchmal gut versteckt sind. Eine andere Möglichkeit ist bei den Computern gegeben. Die IC's, die die serielle Schnittstelle betreiben, lassen sich nämlich durch ein Programm auf verschiedene Datenformate einstellen. Manchmal sind diese Programme schon vorhanden und durch BASIC-Befehle aufrufbar (zum Beispiel beim TI 99/4A — siehe Schnittstellentest in Heft 4).

Die gebräuchlichsten Datenformate:

1 Startbit	7 Datenbits		1 Stopbit
1 Startbit	7 Datenbits		2 Stopbits
1 Startbit	7 Datenbits	1 Paritätsbit	1 Stopbit
1 Startbit	7 Datenbits	1 Paritätsbit	2 Stopbits
1 Startbit	8 Datenbits		1 Stopbit
1 Startbit	8 Datenbits		2 Stopbits
1 Startbit	8 Datenbits	1 Paritätsbit	1 Stopbit

Bei Übertragungsformaten mit Paritätsbit wird noch zwischen gerader ("even") und ungerader ("odd") Parität ("parity") unterschieden.

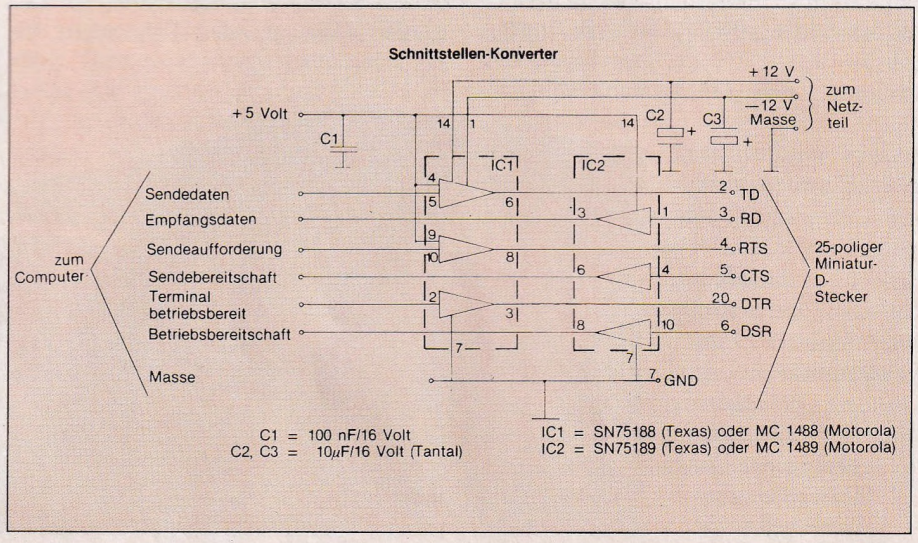
Ein Schnittstellen-Konverter

Wer einen Computer mit einer nicht normgerechten seriellen Schnittstelle hat und ein Gerät mit V.24 Schnittstelle anschließen will, dem kann geholfen werden. Voraussetzung ist allerdings, daß ihr schon einmal einen Lötkolben in der Hand gehabt und die eine oder andere Schaltung erfolgreich aufgebaut habt. Denn einen Kursus im Selbstbau können wir hier nicht geben. Die folgende Schaltung ist recht einfach, aber wer nicht sicher ist, daß er sie richtig aufbauen kann, sollte es lieber lassen.

Der Schnittstellen-Konverter dient dazu, die Signale einer seriellen Schnittstelle auf die Spannungen umzusetzen, die durch die V.24-Norm vergegeben sind. Dazu werden die TTL-Ausgänge an einen Leitungstreiber und die TTL-Eingänge an einen Leitungsempfänger angeschlossen. Für den Leitungstreiber benötigt man ein Netzteil, das +12 Volt und —12 Volt Gleichspannung mit ca. 100 mA (Milliampere) liefern kann. Das solltet ihr euch fertig in einem Elektronikladen kaufen, denn mit Netzspannung ist nicht zu spaßen.

Ein paar Tips noch: verwendet IC-Sockel, achtet bei den Tantal-Kondensatoren auf die richtige Polung und prüft die Schaltung mit einem Ohmmeter, ehe ihr die IC's einsetzt und die Schaltung anschließt. ■

Klaus Liebenwald



Bock auf Block

Ob man die Schuhgrößen seiner Lieblings-Rockstars vergleichen will, das Verhältnis von alten Brötchen zu Hackfleisch in Hamburgern ermittelt oder gerade eine Statistik über die Anzahl der Äh's verschiedener Lehrer pro Schulstunde anfertigt: Mit Blockgraphik klappt das immer.

Wie und was für Daten man auch hat, Blockdiagramme sind oft die übersichtlichste Methode, um Größenvergleiche zwischen ihnen anzustellen. Schön ist, wenn man kein Millimeterpapier zu diesem Unterfangen benutzen muß, sondern der Computer für einen wirbelt. Besonders schön ist es aber, wenn das dazu nötige Programm so kurz und bündig ausfällt, daß man es schon eingegeben hat, während der zeichnende Zeitgenosse noch seinen Bleistift spitzt.

Klaus Weidemann ist für seinen TI ein solcher großer Wurf gelungen, der die Redaktion schier vom Hocker holte. Sogleich gingen unsere Spezis daran, das Listing auch für die anderen Systeme zum Laufen zu bringen. Jeder fühlte sich bei seiner Ehre gepackt und wollte natürlich seine 'Kunden' in den Genuß bringen, nicht nur Bauklötze zu staunen, sondern sie in einem schönen Blockdiagramm auf dem Monitor zu haben. Hier für euch nun die Früchte unserer Mühen.

Texas Instruments

Der TI 99/4A hat hervorragende graphische Eigenschaften, versprach die Werbung, behauptet die Bedienungsanleitung. Für jeden User stellt sich aber die Frage: "Wie nutze ich diese gelobten Eigenschaften?"

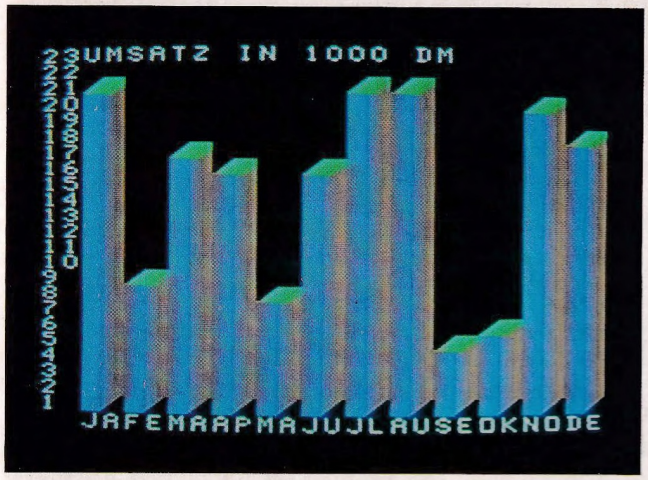
Will man eigene Graphiken aufbauen und nicht die von Texas Instruments im Handbuch vorgegebenen, so stellt sich schnell ein Frusterlebnis nach dem anderen ein. Mit Wut im Bauch und Ärger im Hirn wird die Konsole ausgeschaltet und fortan, wenigstens für ein paar Tage, mit Verachtung gestraft.

Tatsache ist, daß gerade das Gebiet der Graphik mit Dornensträuchern überwuchert ist. Jeder Computer-User wird wissen, wie lange er gebraucht hat, um ein farbiges Erlebnis auf den Bildschirm zu zaubern. Daß Graphik aber nichts mit Zauberei zu tun hat und auch nicht schwer ist, möchten wir euch mit dem folgenden Programm demonstrieren. Graphiken weisen eigentlich immer regelmäßige Strukturen auf. Die Kunst liegt darin, diese Strukturen zu erkennen und geschickt in ein Programm umzusetzen.

Mit der Blockgraphik möchten wir allen TI-Usern eine einfache und schnelle Routine an die Hand geben.
Zum TI-Listing:
Zeile
100 löscht den Bildschirm und definiert ihn schwarz.
110 definiert die Strings A$ und B$
120 baut mit String A$ und B$ fünf Charaktere aus fünf verschiedenen Zeichengruppen auf. Anschließend werden diese fünf Zeichen mit Farbe belegt.
130 erzeugt für die Demonstration eine Zufallszahl.

TEXAS INSTRUMENTS-X-BASIC

```
100 CALL CLEAR :: CALL SCREEN(2)
110 A$="{16*(F)}" :: B$="0103070F1F3F7FFF"
120 CALL CHAR(104,A$,112,A$,120,B$,128,B$,136,B$)::
     CALL COLOR(10,6,2,11,12,2,12,2,12,13,16,2,14,
    12,16)
130 RANDOMIZE
140 FOR I=24 TO 1 STEP -1 :: DISPLAY AT(24-I,1):
    I :: NEXT I
150 DISPLAY AT(1,4):"UMSATZ IN 1000 DM" ::
     CALL COLOR(3,16,2,4,16,2,5,16,2,6,16,2,7,16,2,
    8,16,2)
160 DISPLAY AT(24,4):"JAFEMAAPMAJUJLAUSEOKNODE"
170 FOR P=6 TO 29 STEP 2
180 Z=INT(RND*23)+1
190 CALL VCHAR(24-Z,P,104,Z)::
     CALL VCHAR(24-Z,P+1,112,Z-1)::
     CALL HCHAR(23,P+1,120)
200 CALL HCHAR(24-Z,P+1,136)::
     CALL HCHAR(24-Z,P,128)
210 NEXT P
220 CALL KEY(0,KEY,STAT):: IF STAT=0
    THEN 220 ELSE GOTO 140
```

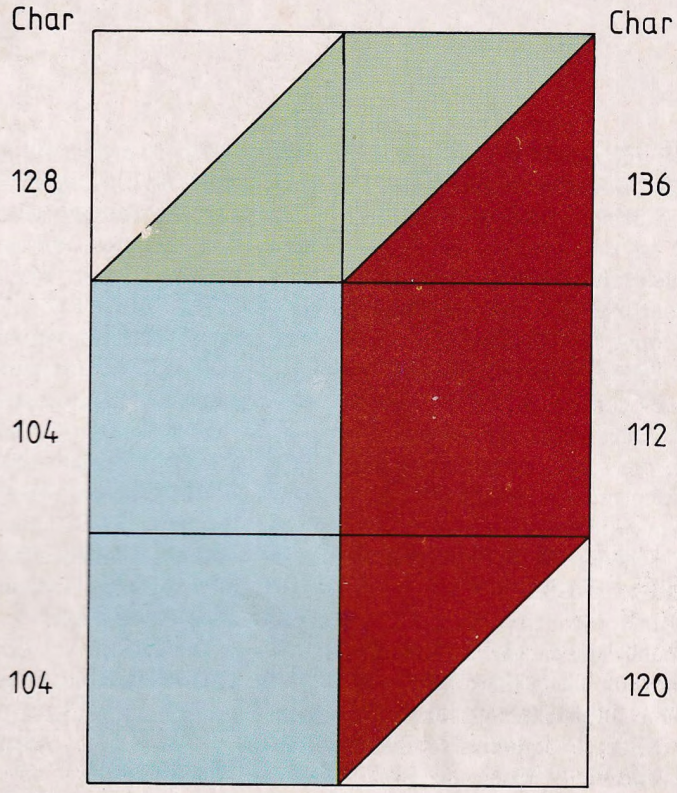


Ob 12 Monate oder 12 Apostel — immer übersichtlich

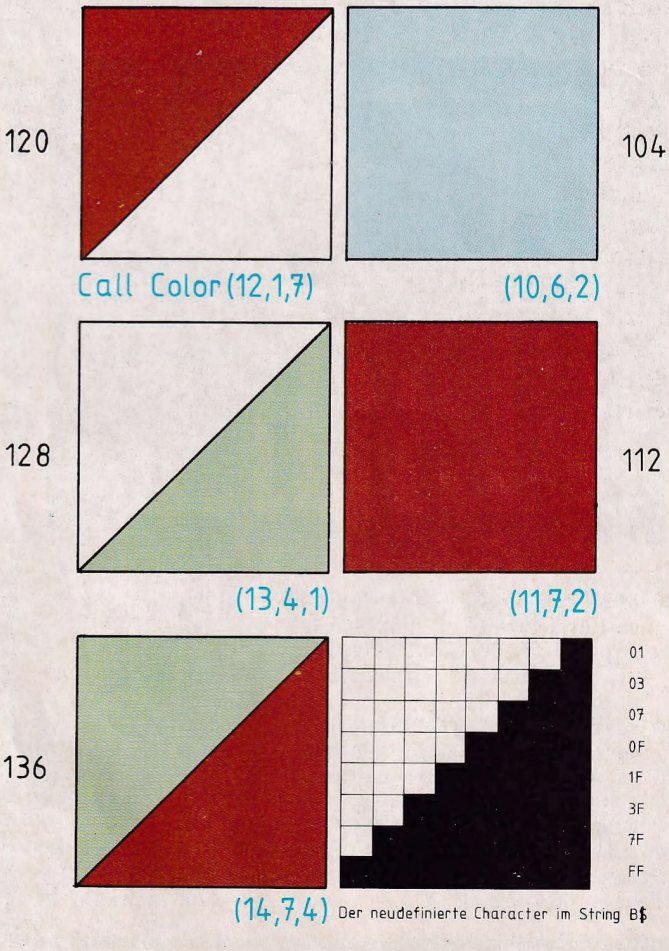
TEXAS INSTRUMENTS-BASIC

```
100 CALL CLEAR
110 CALL SCREEN(2)
120 A$="{16*(F)}"
130 B$="0103070F1F3F7FFF"
140 CALL CHAR(104,A$)
150 CALL CHAR(112,A$)
160 CALL CHAR(120,B$)
170 CALL CHAR(128,B$)
180 CALL CHAR(136,B$)
190 CALL COLOR(10,6,2)
200 CALL COLOR(11,12,2)
210 CALL COLOR(12,2,12)
220 CALL COLOR(13,16,1)
230 CALL COLOR(14,12,16)
240 RANDOMIZE
250 FOR I=23 TO 1 STEP -1
260 PRINT I
270 NEXT I
280 K$="UMSATZ IN 1000 DM"
290 FOR U=1 TO LEN(K$)
300 CALL HCHAR(1,U+7,ASC(SEG$(K$,U,1)))
310 NEXT U
320 FOR U=3 TO 8
330 CALL COLOR(U,16,2)
340 NEXT U
350 D$="JAFEMAAPMAJUJLAUSEOKNODE"
360 FOR U=1 TO LEN(D$)
370 CALL HCHAR(24,U+5,ASC(SEG$(D$,U,1)))
380 NEXT U
390 FOR P=6 TO 29 STEP 2
400 Z=INT(RND*23)+1
410 CALL VCHAR(24-Z,P+1,104,Z)
420 CALL VCHAR(24-Z,P+1,112,Z-1)
430 CALL HCHAR(23,P+1,120)
440 CALL HCHAR(24-Z,P+1,136)
450 CALL HCHAR(24-Z,P,128)
460 NEXT P
470 CALL KEY(0,KEY,STAT)
480 IF STAT=0 THEN 470
490 FOR I=1 TO 23
500 CALL HCHAR(I,6,32,27)
510 NEXT I
520 GOTO 280
```

So werden die Character zusammengesetzt



Farbpaarungen der Character



140—160 erstellt die Texte und die Farbe der Buchstaben, hier weiß. Der Text kann beliebig verändert werden.
170 ist der Anfang der FOR/NEXT-Schleife für den Bildschirm-Aufbau. Die Variable P gibt die Ausgabespalte an.
180 weist der Variablen Z einen, hier für die Demo wieder zufälligen, Wert für die Blockhöhe zu.
190—200 bauen die Blöcke dann auf. Wie die fünf Zeichen dazu angeordnet werden müssen, erseht ihr aus unserem Kästchenschema.
210 schließt die Schleife mit dem notwendigen NEXT.
220 hier wird die CALL KEY-Abfrage gemacht, die die ganze Prozedur auf Tastendruck wiederholt.

Und nun ans Eingemachte! Der Witz an diesem Programm ist, daß für fünf verschiedene Zeichen nur zwei verschiedene Strings nötig sind. Mit der richtigen Farbzuweisung erhalten wir dann einen räumlichen Eindruck.

Wir brauchen fünf verschiedene Zeichen, da es nötig ist, fünf verschiedene Farb-Paare zuzuweisen. Wie und warum erläutert unsere Übersicht. Die Routine kann als Sub-Routine als CALL BLOCK(Z) definiert und dann vom Hauptprogramm angesprochen werden. Die Übergabe-Variable Z wird auch im Hauptprogramm berechnet!

Zum Listing 2 braucht eigentlich nicht mehr viel gesagt werden. Es läuft unter TI-BASIC, im Gegensatz zum ersten, das unter TI-Extended BASIC arbeitet, und hat deshalb keine Mehrfachbefehle in den Zeilen. Dadurch ist es natürlich auch etwas länger.

Die DISPLAY AT-Befehle sind mit CALL HCHAR-Befehlen umschrieben.

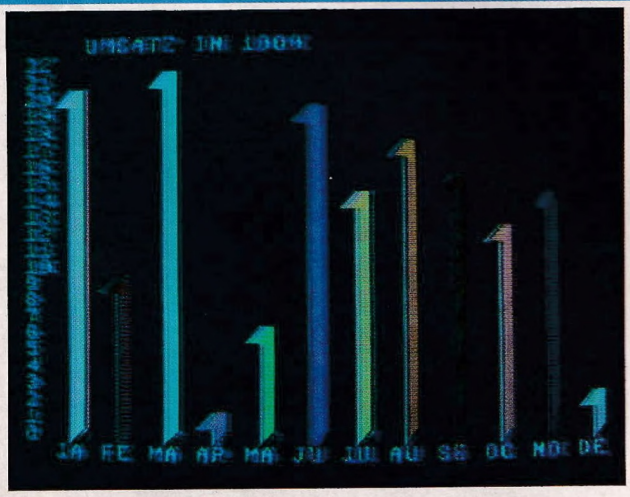
Atari

Doch was dem TI recht, ist den anderen System billig. Zwar wird die Graphik bei Atari, Sinclair und Commodore anders erzeugt, aber das Ergebnis, eine Block-Graphik mit einem Minimum an BASIC, ist das gleiche.
Zum Atari Listing:
Zeile
100—130 Hier wird die Bildschirm-Maske aufgebaut.
140 Zur Demonstration wird eine Basis ermittelt.
150—250 In dieser Schleife werden die Blöcke für den jeweiligen Monat erzeugt. Für die Demo werden die Werte per Zufall ermittelt.
160 Das Programm wird wiederholt, wenn eine Taste gedrückt wird.

CBM 64

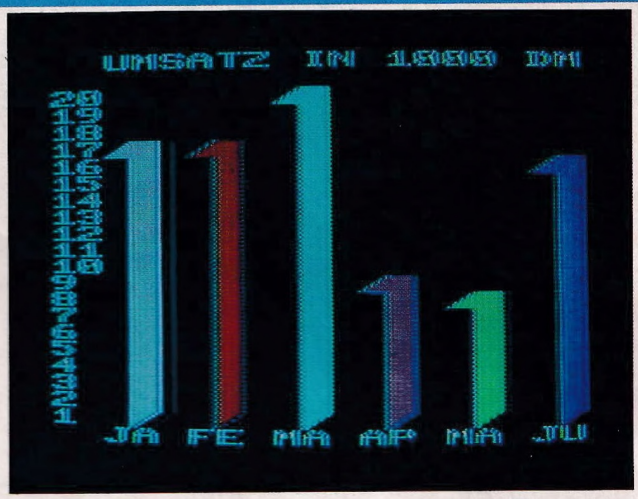
Balkendiagramme auf dem C-64 sind, auch wenn es auf dem ersten Blick schwierig erscheint, ziemlich leicht zu verwirklichen. Man benötigt nicht einmal ein langes, kompliziertes Programm, um dies zu bewerkstelligen.

COMMODORE 64



```
10 a=1948:b=56220:j$="ja fe ma ap ma ju ju au se oc
    no de{HOME}"
20 poke 53280,0:poke 53281,0:
   print "{CLR}      umsatz in 1000":for i=0 to 22:
   print abs(i-22)
30 next :print "{3*RIGHT}"j$:for i=1 to 12:
   w=int(rnd(1)*22):if w=0 and i=12 then 70
40 if w=0 then a=a+3:b=b+3:next
50 poke a,105:poke b,i:for j=1 to w:
   poke a-j*40,160:poke b-j*40,i:next
60 poke a-j*40+39,233:poke b-j*40+39,i:a=a+3:b=b+3:
   next
70 wait 653,4:run
```

COMMODORE VC 20



```
10 j$(0)="ja fe ma ap ma ju":
   j$(6)="ju au se ok no de":for z=0 to 6 step 6:
   a=8145:b=38865
20 poke 36879,8:print "{CLR}{WHT}
     umsatz in 1000 dm{DOWN}":
   for i=20 to 1 step -1:print "{LEFT}"i:next
30 print "{2*RIGHT}"j$(z)"{HOME}":for i=1 to 6:
   w=int(rnd(1)*20):if w=0 and i=6 then 70
40 if w=0 then a=a+3:b=b+3:next
50 poke b,i:poke a,105:for j=1 to w:poke b-j*22,i:
   poke a-j*22,160:next
60 poke b-j*22+21,i:poke a-j*22+21,233:a=a+3:b=b+3:
   next
70 wait 653,4:next :run
```

Peek und Poke

Es gibt etwas, das es eigentlich gar nicht gibt!

Wenn wir als eingefleischte TI 99/4A-User hin und wieder die Listings von Programmen für andere Computer ansehen, kann man manchmal direkt neidisch werden. Sie strotzen oft vor lauter Peek's und Poke's. In manchen Fällen läßt sich über die Nützlichkeit durchaus streiten, aber es gibt immer wieder Gelegenheiten, bei denen ein Peek oder Poke einen gewissen Geschwindigkeitsvorteil bietet. Ganz abgesehen von 'Befehlen', die im BASIC nicht angeboten werden. Mit Peek können wir bekanntlich den Wert einer bestimmten Speicherstelle auslesen, während mit Poke ein bestimmter Wert in eine Speicherstelle geschrieben werden kann.

Flotte Sprites

Texas Instruments hat beim TI 99/4A aber leider eine Grenze eingebaut: Um mit Peek, bei TI CALL PEEK genannt, und Poke, bei TI CALL LOAD, arbeiten zu können, benötigen wir entweder Extended BASIC oder das Mini-Memory. Letzteres enthält noch zwei weitere interessante Befehle, nämlich CALL PEEKV und CALL POKEV. Das V am Ende der Befehle deutet schon an, daß sie zum Schreiben bzw. Lesen von Werten aus dem Video-RAM bestimmt sind.

Dieses Video-RAM wird beim TI 99/4A von einem Video-Display-Prozessor (VDP) verwaltet und enthält alle Daten, die zur Bildschirmdarstellung benötigt werden. Dazu zählen Bildschirminhalt, Farbinformationen, Charakterdarstellung und alles über Sprites. Als Besonderheit kommt beim TI 99/4A noch das komplette BASIC-Programm (bei TI-BASIC und bei Extended BASIC ohne Speichererweiterung) hinzu. Ist unter Extended BASIC eine Speichererweiterung angeschlossen, so befinden sich im VDP-RAM vom BASIC-Programm nur noch die Stringvariablen und die Variablenlisten.

Bevor wir an ein paar praktische Beispiele gehen können, sind noch zwei Erklärungen fällig. Zum ersten handelt es sich dabei um die Adressierungsart in den Befehlen. Die Adresse muß als dezimale Zahl angegeben werden, die, wenn sie über 32767 (Hex > 7FFF) liegt, negativ ist. Um also eine Adresse über 32767 ansprechen zu können, müssen wir 65536 von ihr abziehen. Zum anderen benötigen wir eine Speichererweiterung, um in Extended BASIC CALL LOAD durchführen zu können. Dies scheint auf den ersten Blick etwas verwunderlich zu sein, aber der Befehl CALL LOAD kann nicht nur Werte poken, sondern auch ganze Maschinenprogramme von Diskette laden. Hierzu werden mit dem Modul eine Reihe von Hilfsroutinen in die Speichererweiterung geladen, was denn auch die Durchführung dieses Befehls vor CALL LOAD erforderlich macht. CALL PEEK kann dagegen auch ohne Speichererweiterung durchgeführt werden.

Genug der Theorie, zumal die Handbücher zu den Modulen auch noch Auskunft über die Funktion der Befehle geben. Für heute bleiben wir ohnehin beim Extended BASIC Modul mit Speichererweiterung, wobei — wie erwähnt — alle CALL PEEK Befehle aber auch ohne arbeiten.

Ein ganz besonders interessanter Speicherbereich beim TI 99/4A ist der von Texas Instruments als Scratch-Pad-RAM bezeichnete Adressenbereich von > 8300 bis > 83FF bzw. dezimal —32000 bis —31745. In diesem Bereich lohnt es sich zu suchen und zu experimentieren. Aber Vorsicht: Falsche Werte können hier sehr schnell zu einem Absturz des Systems führen.

Insbesondere bei Spielprogrammen, die Sprites nutzen, können die beiden Speicherstellen > 837A und > 837B (—31878 und —31877) wirkungsvoll eingesetzt werden. In > 837B befindet sich nämlich immer eine Kopie des VDP-Statusregisters, in dem ständig ein ganz bestimmtes Bit gesetzt ist, wenn sich die Punkte von zwei oder mehreren Sprites überlappen, so daß sich damit eine Trefferabfrage realisieren läßt. CALL

COINC ist demgegenüber sicherlich langsamer, allerdings auch mächtiger, denn dadurch lassen sich einzelne Angaben eindeutig definieren.

Ein kurzes Beispielprogramm:

```
100 REM COINCIDENCE UEBER
    VDP-STATUS
110 CALL CLEAR
120 CALL MAGNIFY (2)
130 CALL SPRITE (#1,42,2,86,162)
140 GOSUB 180
150 CALL SPRITE
    (#2,42,2,170,156,50,0)
160 GOSUB 180
170 GOTO 160
180 CALL PEEK (—31877,A)
190 IF (A AND 32)=32 THEN
    B=B+1 ::
    PRINT "TREFFER NR.";B
200 RETURN
```

Zu Beginn werden erst einmal 2 Sprites erschaffen, wovon einer eine bestimmte Position auf den Bildschirm einnimmt, während der andere sich von oben nach unten bewegt. Die Trefferabfrage wurde als Unterprogramm ausgeführt, damit sie einfacher in andere Programme eingebunden werden kann. Zei-

le 180 fragt mit CALL PEEK die Speicherstelle > 837B ab und übergibt den Wert an die Variable A. Der Wert selbst ist im Prinzip ohne Bedeutung, da nur interessant ist, ob ein bestimmtes Bit auf 1 gesetzt ist. Dies wird mit einem AND-Vergleich in der IF THEN Abfrage ermittelt. Jedesmal, wenn der eine Sprite den anderen trifft, erfolgt auf der linken Seite des Bildschirms eine Trefferanzeige. Nochmal kurz zur Erinnerung: Diese Abfrage darf nur eingesetzt werden, wenn entweder nur zwei Sprites auf dem Bildschirm sind, oder die zwei Sprites, die kollidieren können, eindeutig bekannt sind. Ansonsten zeigt diese Abfrage jeden Treffer aller Sprites untereinander an!

Auch die Bewegung von Sprites kann über direkten Zugriff auf das Scratch-Pad-RAM gesteuert werden. Die Speicherstelle > 837A enthält die Nummer des höchstnumerierten Sprites in Bewegung. Wird dieses Byte 0 gesetzt, wird schlagartig die Bewegung aller Sprites gestoppt. Liegt der Wert z.B. bei 5, so werden alle Sprites mit Nummern über 5 gestoppt. Das folgende Programm zeigt die mögliche Verwendung dieser Speicherstelle, wobei vor Programmstart einmal CALL INIT durchgeführt werden muß:

```
170 GOSUB 300
180 FOR X=10 TO 0 STEP -1
190 CALL LOAD (-31878,X)
200 GOSUB 300
210 NEXT X
220 FOR X=1 TO 10
230 CALL LOAD (-31878,X)
240 GOSUB 300
250 NEXT X
260 CALL LOAD (-31806,64)
270 GOSUB 300 :: GOSUB 300
280 CALL LOAD (-31806,0)
290 GOTO 180
300 FOR Y=1 TO 200 :: NEXT Y ::
    RETURN
```

Zehn Sprites mit zufällig generierten Geschwindigkeiten werden geschaffen und dann nacheinander über > 837A gestoppt und wieder in der Bewegung freigegeben. Mit CALL LOAD (-31806,64) werden nochmals alle angehalten und wiederum freigegeben. > 83C2 bzw. -31806 bleibt von irgendwelchen Sprite-Befehlen unbeeinflußt, so daß sich diese Speicherstelle gut dazu eignet, mehrfarbige Sprites zu bewegen. Mehrfarbige Sprites müssen selbstverständlich aus diversen einzelnen Sprites zusammengesetzt werden.

Bei der gleichförmigen Bewegung dieser Gebilde gibt es im BASIC aber einige Schwierigkeiten. Auch hierfür ein einfaches Beispiel:

```
100 REM MEHRFARBIGE SPRITES
110 CALL CLEAR
120 CALL CHAR
    (128,"FFFFFFFFFFFFFFFF")
130 CALL LOAD (-31806,64)
140 CALL SPRITE
    (#1,128,5,80,80,0,40)
150 CALL SPRITE
    (#25,128,9,72,80,0,40)
160 CALL LOAD (-31806,0)
170 FOR X=1 TO 1000 :: NEXT X
180 CALL LOAD (-31806,64)
190 FOR X=1 TO 400 :: NEXT X
200 IF B=0 THEN B=B+1 ::
    GOTO160
210 CALL LOAD (-31806,0)
220 END
```

Fügen wir in dieses Programm eine Zeile 125 GOTO 140 ein, werden die Schwierigkeiten, die in BASIC mit mehrfarbigen Sprites auftreten können, deutlich. Deshalb wird vor der Definition der Sprites jegliche Spritebewegung über > 83C2 unterbunden. Erst wenn die Sprites und alle einzelnen Geschwindigkeiten bestimmt sind, wird die Bewegung wieder freigegeben. Da > 83C2 entgegen > 837A vom BASIC aus nicht beeinflußt wird, birgt diese Speicherstelle aber auch gewisse Gefahren: So bewegen sich nach CALL LOAD (-31806,64) keine Sprites mehr, auch wenn neue Programme geladen wurden; erst ein BYE bzw. die Quit-Taste löscht das Byte.

Ja, die leidige Quit-Taste! Auch sie läßt sich über > 83C2 abschalten. Einfach CALL LOAD (-31806,16) eingeben, und das unabsichtliche Erscheinen des Titelbildes hat ein Ende. Sollen die automatische Spritebewegung und die Quit-Taste abgeschaltet werden, so sind die entsprechenden Werte zu addieren, d.h. es ist CALL LOAD (-31806,80) einzugeben.

Das BASIC-Flag-Byte

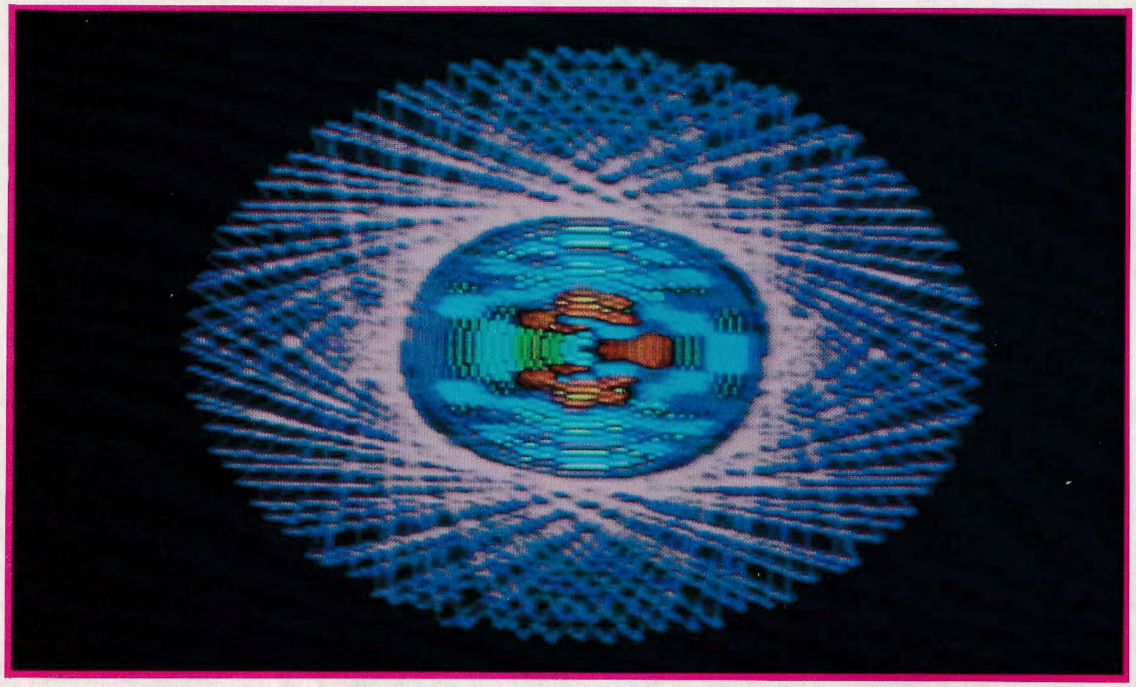
Zum Schluß für heute wollen wir uns noch mit der Speicherstelle > 8345 beschäftigen, die auch als BASIC-Flag-Byte bezeichnet werden kann. Hier sind zwar bitweise einige Informationen abgespeichert, die während des Ablaufs eines BASIC-Programmes benötigt werden, im Wesentlichen können aber nur BASIC-Befehle nachgeahmt werden:
CALL LOAD (-31931,2) simuliert ON WARNING NEXT.
CALL LOAD (-31931,4) ersetzt ON WARNING STOP.
CALL LOAD (-31931,16) ist TRACE.
CALL LOAD (-31931,64) ersetzt ON BREAK NEXT.
CALL LOAD (-31931,128) setzt den Listschutz.
CALL LOAD (-31931,0) hebt alle eben genannten Wirkungen auf.

So, das war's für heute und nun viel Spaß beim Ausprobieren der Peek's und Poke's beim TI 99/4A! ■

Heiner Martin

Von Affen und Verwandten

High-Res-Graphik und Sprite Editor im Härtetest lehrten uns, den TI 99/4A wieder zu lieben.



Gute Software scheint für den TI 99/4A kaum zu existieren. Einfache, mehr oder weniger niedliche BASIC- oder Extended-BASIC-Spiele waren und sind noch die Hits unter vielen TI-Usern. Doch so richtige Freude kommt beim langsamen TI BASIC nicht auf.

Aber jetzt gibt es Apesoft, und dieser Affe hat es in sich. Jetzt können alle TI-User aufatmen. Der TI, bestückt mit dem Apesoft-Graphikpaket, zeigt jetzt den Commodores, Sinclairs oder wie sie alle heißen, seine 16-Bit-Processor-Auspuffrohre.

Das Apesoft-Graphikpaket beinhaltet, wie der Namen schon sagt, reine Graphik. Das Schlüsselwort lautet High Resolution.

Da ist zuerst der Apesoft Bit Map Mode Graphic Generator/Interpreter. Mit diesem Paket kann der gesamte Bildschirm genutzt werden. 256x192 Pixel können beliebig auf dem Fernsehschirm angeordnet werden. Selbstverständlich in allen 16 Farben.

Also: die Diskette ins Laufwerk und das Programm gestartet. Aber schon fangen die Schwierigkeiten an. Wie war doch gleich die Syntax? Und wie lautet der Programmname? Apesoft scheint eine Vorliebe für gehässig-schwierige Programmnamen zu haben. Gottseidank findet sich alles im gut aufgemachten und durchdachten Manual wieder. Doch das ständige Nachschlagen und Blättern zerrt an den Nerven.

Apesoft Bit Map Mode: mit affenartiger Geschwindigkeit

Hat man dann mit vor Aufregung zitternden Fingern und grinsenden Commodore-Freaks im Nacken das Programm gestartet, erfolgt eine weitere Geduldsprobe! Das Laden der Maschinenprogramme dauert entschieden zu lange. Und die Commodore-Herren grinsen immer noch. Doch dann legt die Demo los. Erst ganz langsam, doch dann plötzlich sehr schnell und farbig. (Die Commodore-Herren hatten plötzlich keine Farbe mehr, das Grinsen erstarb.) Was man da zu sehen bekam, konnte einen wirklich umhauen. High-Res-Graphik in allen Varianten. Kreise, Blumen, Rechtecke, Säulen, 3-D-Kurvendiagramme, Ellipsen etc., und alles schillert in 16 Farben. Selbst komplizierte Kurven produziert der ach so langsame TI 99/4A mit einer rasanten Geschwindigkeit, die eben nur ein 16-Bit Prozessor vorlegen kann. Diese Software stammt von Profis!

Mit gleicher Sorgfalt wie die Demo wurden die beiden Graphikpakete programmiert. 40 Graphik-Befehle im Bit Map Mode lassen keine Wünsche offen. Alle Befehle sind kinderleicht in ganz normale BASIC-Routinen einzubauen und funktionieren anstandslos. Und wirklich schnell. Selbst für das Abspeichern der fertig erstellten Graphiken gibt es spezielle Befehle, und wenn die Graphik gedruckt werden soll, ist das auch kein Problem. Das Graphik-Paket besitzt eine eingebaute Druckerroutine für Epson FX-80, RX-80, die wirklich narrensicher funktioniert.

Doch wo Licht ist, ist auch Schatten. Um das Graphik-Paket von Apesoft nutzen zu können, braucht ihr eine Mindestausstattung an Geräten:
Konsole TI 99/4A

+ Editor-Assembler-Modul oder Extended-BASIC Modul
+ 32 KRAM Expansion
+ Disketten-Station

Selbstverständlich kann der Drucker zusätzlich angeschlossen werden.

Die 32 KRAM Expansion wird bekannterweise auf dem Markt angeboten, aber wo bleiben der Floppy-Controller und das Laufwerk? Ohne Floppy keine Graphik!

XG-Basic: empfehlenswert

Wer sich in das Manual eingelesen hat und allmählich die Graphikbefehle auswendig kennt, der wird die Turtle-orientierte Bit Map Graphik lieben lernen.

Das zweite Graphik-Paket ist das Expanded Graphic BASIC oder kurz XG-BASIC. Auch hier stehen 40 Graphikbefehle zur Verfügung, die über BASIC-Routinen aufgerufen werden. Kinderleicht. Eine einfache FOR-NEXT-Schleife erzielt schon erstaunliche Ergebnisse.

Die Druckerausgabe ist genauso eingebaut wie im Bit Map Paket und ebenso leicht zu bedienen. Die Ausführungsgeschwindigkeit von XGBASIC läßt keine Wünsche offen — dafür aber die Münder der eben noch lästernden Commodore-Leute.

Es ist schon eine Freude, beim Bildaufbau des XGBASIC-Graphikpakets zuzusehen. Die Auflösung beträgt auch hier 256x192 Pixel in allen 16 Farben, die der TI kennt.

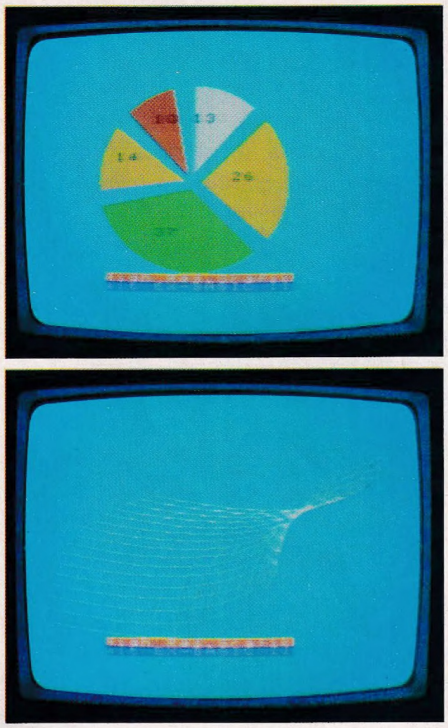
Auch bei diesem Graphikpaket muß man eine Mindestkonfiguration an Geräten zur Verfügung haben:

Konsole TI 99/4A
+ Mini Memory
+ Diskettenstation oder Cassettenrecorder

oder

Konsole TI 99/4A
+ 32 KRAM Expansion
+ Editor Assembler oder Ex BASIC
+ Diskettenstation

Für die Mini Memory Benutzer gibt es aufgrund der geringen Speicherkapazität des Moduls einige Einschränkungen. Es entfallen zum Beispiel



EX-BASIC II im Einsatz zeigt Beachtliches

Hardcopy-Befehle, Bildabspeicherung auf Diskette und Ähnliches. Von den Graphikmöglichkeiten entgeht dem Mini Mem allerdings nichts.

Das XGBASIC teilt den Bildschirm in 4 Windows auf. Nur in diesen Windows kann eine Graphik generiert werden. Doch keine Sorge. Das Window ist mit 128x120 Pixel ausreichend groß dimensioniert, kann beliebig auf dem Bildschirm plaziert und auch vervielfältigt werden. Außerdem kann Text mit der Graphik ausgegeben werden.

Leider funktionieren einige schöne Standardroutinen des TI nicht mehr, wenn eines der beiden Graphikpakete geladen ist. CALL HCHAR, VCHAR, DISPLAY, PRINT und CALL SPRITE sind nicht mehr zulässig. Doch wer es auf schöne Graphik absieht, dafür aber auf wenig Text, der wird sich daran nicht stören. Etwas ärgerlicher ist da schon die lange Ladezeit und die notwendige Vorsicht im Umgang mit dem "New". Teile der Apesoft Programme werden im High-Memory Bereich gespeichert und jedes "New" räumt dort oben kräftig auf, was den Verlust von Befehlen bedeutet. Beim Mini Memory Modul tritt diese Schwäche nicht auf.

Um euch zu verdeutlichen, was die Apesoft Graphikpakete alles können, möchten wir einige Befehle vorführen.

Der Befehl CALL LINK ("Circle, X, Y, R) zeichnet einen Kreis mit dem Mittelpunkt X, Y und dem Radius R.
CALL LINK ("Setto", X, Y) setzt einen Pixel an die Stelle X, Y.
CALL LINK ("Ellipsis", X, Y, A, B) zeichnet Ellipsen mit dem Mittelpunkt X, Y, großer Halbachse A und kleiner Halbachse B.

Apesofts Pakete: Spitze aber teuer

Genauso einfach können mit einem Befehl ganze Rechtecke, Blockdiagramme oder Säulen erstellt werden.

Ohne Umschweife: Die Apesoft Graphikpakete sind das derzeit beste, was uns auf dem Markt bekannt ist. Trotz der Einschränkungen beim XGBasic kann diese Software allen TI-Usern wärmstens ans Herz gelegt werden.

Aber warum muß gute TI-Software immer so teuer sein? Jedes Apesoft-Paket kostet deutlich über 200 Mark, und so viel Geld kann wirklich nicht jeder TI-User erübrigen.

Wenn's nicht ganz so teuer sein soll: EX-BASIC II

Doch laßt die Köpfe nicht hängen. Auch für den etwas schmaleren Geldbeutel gibt es schnelle Graphik in Maschinensprache. 148 Mark sind zwar auch noch kein Pappenstiel, aber dafür bietet Herr Kull mit seinem EX-BASIC II Graphikprogramm auf Diskette durchaus Sehenswertes!

Wie die Apesoft-Pakete voll in Assembler geschrieben und entsprechend schnell bietet Ex-Basic II nach dem Laden eine sehr hübsche Überraschung. Der TI verfügt plötzlich über echte Kleinbuchstaben und Umlaute. Das wird man bei Apesoft lange suchen müssen. Eine weitere Raffinesse stellt die Softwareuhr dar, die sehr einfach (im Direktmodus) in der rechten oberen Bildschirmhälfte eingebaut werden kann. Und die Uhr läuft genau!

Wie bei Apesoft wurden auch hier alle Graphikbefehle in BASICprogramme

eingebaut und mit CALL LINK ("...") aufgerufen. Die Ausführungsgeschwindigkeit ist nicht ganz so hoch wie bei den Apesoft-Paketen. Leider ist auch hier (wie bei Apesoft XGBASIC) nicht der gesamte Bildschirm nutzbar. Die obere und die untere Bildschirmzeile können von der Graphik nicht belegt werden. Dafür aber alle 16 TI-Farben.

EX BASIC II läßt alle 28 Sprites zu, die jedoch bis auf CALL MAGNIFY als Originalbefehle im Graphikmodus nicht wirksam sind. Es wurden deshalb eigene Unterprogramme entwickelt, die sehr sicher funktionieren.

Leider besitzt EX BASIC II keine Hardcopyroutine, wohl aber eine Bildschirmspeicherung auf Diskette, die vom BASICprogramm aus gesteuert wird. EX BASIC II kennt PEEKV, POKEV und auch PRINT, so daß Text im Graphikmodus verwendet werden kann. Die Befehle SET (um einen Punkt zu setzen), DRAW (für eine Linie) und CIRCLE sowie PAINT (zum Ausfüllen von Flächen) genügen schon, um sehr schöne Graphiken zu erstellen. Über den CIRCLE-Befehl lassen sich aber auch Ellipsen oder Kreisbögen generieren.

Wer sagt denn, daß der TI 99/4A für Graphik zu langsam ist?

Aber auch hier gibt es Schatten ... EX BASIC II erwies sich als etwas störanfällig. Bei der Erprobung kam es des öfteren zu unmotivierten Systemabstürzen. Böse, wenn sich ein mühsam erstelltes BASICprogramm mit Top-Graphik, nicht auf Diskette gesichert, plötzlich ins Nirwana verabschiedet.

Wer einen Systemabsturz provozieren will, der schafft das auch. Ein BREAK mit sofortigem REDO hat meinen TI mehrmals über den Jordan geschickt.

Ein weiterer Kritikpunkt ist das Manual. Zwar deutlich und klar geschrieben, aber die äußere Form verdient doch Tadel. 8 Seiten Fotokopie im Format DIN A 4, keine besonders gute Qualität, einfach geklammert. Ein einfacher Schnellhefter würde hier schon mehr hermachen, und so ein Hefter geht auch nicht so leicht verloren!

Trotz der Kritik an EX BASIC II kann auch diese Software mit ruhigem Gewissen weiterempfohlen werden, zumal der Kaufpreis von 148 Mark zu den gebotenen Leistungen in einem durchaus ausgewogenen Verhältnis steht. Und wenn Herr Kull noch den allerletzten Feinschliff vollzieht, kann EX BASIC II durchaus als ernstzunehmende Konkurrenz neben den Apesoft-Programmen bestehen.

Extended Sprite Designer: nicht der Wahnsinn

Als letztes wollen wir euch noch das Graphikpaket mit dem Namen Extended Sprite Designer vorstellen. Dieses Päckchen wird auf zwei Cassetten geliefert und kostet 40 Mark. Der Sprite Designer läuft unter Extended BASIC und soll zwei Anwendungen haben:
1. den fortgeschrittenen Programmierer beim Erstellen und Austesten bewegter Graphik unterstützen
2. dem Programmieranfänger das Erstellen bewegter Graphik überhaupt ermöglichen.

Als erstes solltet ihr die sechs Seiten Programmerklärung lesen, die recht ausführlich das Programmpaket beschreiben. Dann das Hauptprogramm laden (Cassette mit dem Aufkleber HP).

Zum Erstellen steht ein umrahmtes Bildschirmviertel zur Verfügung. Rechts daneben verschiedene Kürzel, die euch die Arbeit erleichtern sollen. In der Mitte des umrahmten Feldes befindet sch noch ein kleines Kreuz, das über die Cursortasten bewegt werden kann. Um einen Punkt zu setzen, wird einfach gedrückt, und schon ist ein dicker schwarzer Kasten im Feld zu sehen. Auch Linien können innerhalb des Feldes gezogen werden. Mit etwas Phantasie dürfte eine simple Figur zu schaffen sein.

Wenn der Sprite fertig ist, kann durch Drücken der Taste A der entsprechende Hex-Code in Erfahrung gebracht werden. Oder ihr speichert den Sprite über die M-Taste einfach ab. Das Programm besitzt nämlich einen eigenen Speicher, der bis zu 64 Sprites aufnehmen kann. Trickfilmsequenzen können durchaus so hergestellt werden. Über Taste L können dann die Sprites sogar auf Band aufgezeichnet werden, um später wieder verwendet zu werden.

Das Austesten der Sprites ist eine einfach zu handhabende Sache. Spritenummer, Anzahl der Bewegungen und deren Dauer werden angegeben, dazu die entsprechende Spritenummer gedrückt. Der angesprochene Sprite wird dann weiß und marschiert los. Die Bewegungsgeschwindigkeit der Sprites kann verändert werden.

Selbstverständlich kann der Sprite Designer eure selbsterstellten Graphiken auch in eure eigenen Programme einbinden. Dazu gibt es die Unterprogramme (Cassette mit Aufkleber UP). Und das war dann auch schon alles, was man für die 40 Mark bekommt.

Wem's gar nicht recht zu machen ist, der bleibe halt bei Bleistift und Papier.

Der Sprite Designer arbietet sehr langsam und ist meiner Meinung nach für Besitzer des Extended Basic Moduls überflüssig. Hat doch das Modul so schöne Befehle wie CALL SPRITE, CALL MOTION, CALL PATTERN und CALL MAGNIFY. Der fortgeschrittene Programmierer wird sich mit Sicherheit dieser Befehle bedienen, um seine Sprites zu erstellen und zu testen. Die dazu notwendige Routine ist schnell gebaut. Und der blutige Laie wird keine Programme bauen können, die die über den Sprite Designer erstellten Sprites einbinden könnten.

Mit der Original TI-Ausrüstung können 28 Sprites zugleich dargestellt werden, der Sprite Designer vernichtet 20 davon und stellt nur die restlichen 8 auf dem Bildschirm dar. Die Bedienung des Sprite Designers empfinden wir als unzumutbar und zu kompliziert. Zumal wir schon Sprite Editoren kennen, die einfacher zu bedienen und schneller waren (abgetippt aus Zeitschriften und Büchern!) Allerdings haben wir noch nie einen Sprite-Editor benötigt! Bleistift und Papier schaffen das gleiche, sind schneller und fördern das Verständnis der Arbeitsweise des Computers. ■

K. W.

PYRAMIDE

Mit Thorsten Köster in die BASIC-Wüste

Der Pharao hat beschlossen, umzuziehen. Ihr, Baumeister von Ra's Gnaden, seid damit beauftragt, die aus sechs Bauteilen bestehende Pyramide an ihren neuen Standort zu transportieren.

Jeder erfahrende Architekt weiß, daß man niemals ein größeres Bauteil auf ein kleineres legen darf, also wird ein Zwischenlager nötig sein.

Für das ganze Unternehmen habt ihr 80 Tage Zeit, die einzelnen Steinblöcke sind aber so riesig, daß ihr pro Tag nur einen einzigen bewegen könnt.

Der große Pharao hat versprochen, den Baumeister in Gold aufzuwiegen, wenn das Unternehmen gelingt. Sollte er aber an der Aufgabe scheitern, dann wird er den Krododilen zum Frühstück vorgeworfen.

Variablenliste

I	Schleifenvariable
C	Anzahl Tage
T	Tonfrequenz
ST	Statusvariable
K	Inputvariable für den alten Stapel
KE	Inputvariable für den neuen Stapel
JA	untere Zeile des alten Stapels
JE	obere Zeile des alten Stapels
S	Spaltenposition alter Stapel
E	Rückgabevariable
W	Anzahl Wiederholungen
ZE	untere Zeile des neuen Stapels
ZA	obere Zeile des neuen Stapels
A	Rückgabevariable
SP	Spaltenposition neuer Stapel
V	Rückgabevariable
U	Rückgabevariable
P	Schleifenvariable
KEY	Inputvariable Spielregelanzeige
Q	Zeilenposition Graphik
X	Spaltenposition Graphik
Y	Zeichen

Zeichendefinitionen

65 70	Pyramidenteile
72 79	Bohrturm
88 90	Palmenstamm
91 95	Palmenblätter
35 37	Krokodile
80	Wasser

Programmstruktur:

Zeile 160—500: Spielvorbereitung. Fast alle Großbuchstaben werden für die Graphik umdefiniert, weshalb alle Spielinstruktionen mit kleinen Buchstaben zu schreiben sind.

Zeile 510—650: Eingabe des Stapels, von dem ein Pyramidenteil entfernt werden soll, sowie des neuen Stapels. Um das Spiel benutzerfreundlich zu gestalten, erfolgt die Eingabe über das KEY-Unterprogramm, so daß kein ENTER zu drücken ist.

Zeile 660—820: Über das GCHAR-Unterprogramm wird festgestellt, welches Pyramidenteil die Spitze des alten Stapels bildet.

Zeile 830—990: Das gleiche Unterprogramm überprüft, ob der eingegebene Zug gültig ist. Ein gültiger Zug wird ausgeführt, ansonsten folgt erneut eine Eingabeaufforderung.

Zeile 1000—1080: Kontrolle des Spielendes und Verzweigung zu den entsprechenden Schlußtexten und Graphikeffekten.

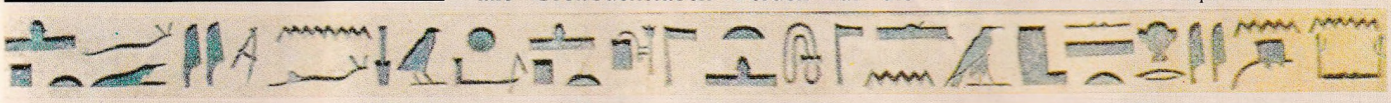
Zeile 1090—1140: Printroutine und Sound zur Eingabeaufforderung.

Zeile 1090—1440: Zuweisung der Stapelkoordinaten.

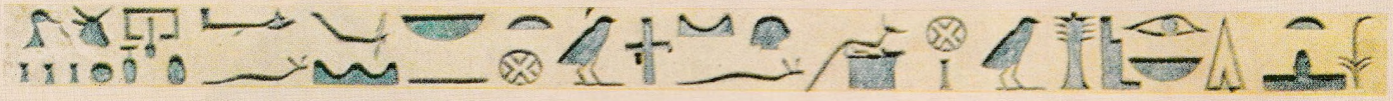
Zeile 1410—1500: Printroutine für die Anzahl der Züge.

Zeile 1510—1970: Titel, Instruktionen und Schlußtexte.

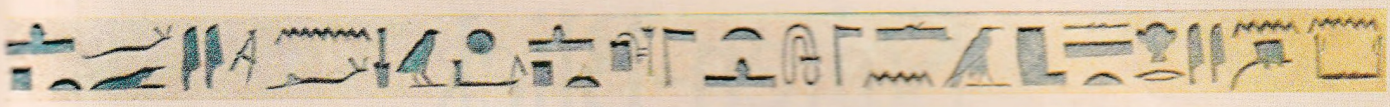
Zeile 1980—2460: Graphikaufbau.



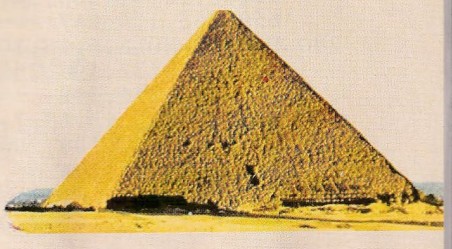
```
100 REM  P Y R A M I D E{11*(SPACE)}
110 REM  THORSTEN KOESTER{15*(SPACE)}
120 REM  SIEDLUNGSWEG 9{15*(SPACE)}
130 REM  3405 ROSDORF{17*(SPACE)}
140 DIM Q(124),X(124),Y(36)
150 GOSUB 1520
160 REM CHAR DEFINITION
170 CODE$="007E7E7E7E7E7E00"
180 FOR I=65 TO 70
190 CALL CHAR(I,CODE$)
200 NEXT I
210 CALL CHAR(72,"0101010102020202")
220 CALL CHAR(73,"8080808040404040")
230 CALL CHAR(74,"0305050509090909")
```



```
240 CALL CHAR(75,"2020202010101010")
250 CALL CHAR(76,"1F{6*(1)}21212121")
260 CALL CHAR(77,"F808080804040404")
270 CALL CHAR(78,"7F414141{8*(F)}")
280 CALL CHAR(79,"FE020202{8*(F)}")
290 CALL CHAR(80,"54AA110044AA1100")
300 CALL CHAR(88,"18181818183C7EFF")
310 CALL CHAR(89,"1818181818181818")
320 CALL CHAR(90,"FDFEDF9F1B191818")
330 CALL CHAR(91,"00C0E0603133B6FC")
340 CALL CHAR(92,"000307{6*(0)}3FFF")
350 CALL CHAR(93,"3F0303070E1C3040")
360 CALL CHAR(94,"003E71E0C08078FC")
370 CALL CHAR(95,"FF01000080C0C020")
380 CALL CHAR(35,"{7*(0)}F7F800000")
390 CALL CHAR(36,"0000{6*(F)}422142")
400 CALL CHAR(37,"01F6DCF8F80C0601")
410 CALL HCHAR(1,1,127,768)
420 CALL SCREEN(12)
430 CALL COLOR(5,2,1)
440 CALL COLOR(7,5,8)
450 CALL COLOR(1,13,1)
460 CALL COLOR(8,13,1)
470 GOSUB 1980
480 C=1
490 C$=STR$(C)
500 GOSUB 1460
510 REM DATENEINGABE
520 T$="von wo?{12*(SPACE)}"
530 T=800
540 GOSUB 1090
550 CALL KEY(0,K,ST)
560 IF ST=0 THEN 550
570 IF (K<>49)=(K<>50)=(K<>51)THEN 550
580 ON K-48 GOSUB 1150,1200,1250
590 T$="nach wo?{11*(SPACE)}"
600 T=800
610 GOSUB 1090
620 CALL KEY(0,KE,ST)
630 IF ST=0 THEN 620
640 IF (KE<>49)=(KE<>50)=(KE<>51)THEN 620
650 ON KE-48 GOSUB 1300,1350,1400
660 REM  AUFFINDEN DER{15*(SPACE)}
    SPITZE DES ALTEN{12*(SPACE)}STAPELS
670 FOR J=JA TO JE
680 CALL GCHAR(J,S,E)
690 IF E<>127 THEN 710 ELSE 700
700 NEXT J
710 IF E<>65 THEN 730
720 W=1
730 IF E<>66 THEN 750
740 W=3
750 IF E<>67 THEN 770
760 W=5
770 IF E<>68 THEN 790
780 W=7
790 IF E<>69 THEN 810
800 W=9
810 IF E<>70 THEN 840
820 W=11
830 REM  ABFRAGEN UND{16*(SPACE)}
    ANZEIGEN NEUER{14*(SPACE)}STAPEL
840 FOR Z=ZE TO ZA STEP -1
850 CALL GCHAR(Z,SP,A)
860 IF A<>127 THEN 870 ELSE 880
870 NEXT Z
880 CALL GCHAR(Z+1,SP,V)
890 IF E>=V THEN 900 ELSE 940
900 T$="das geht nicht...  "
910 T=300
920 GOSUB 1090
930 GOTO 490
940 CALL HCHAR(J,S-5,127,11)
950 CALL HCHAR(Z,(SP-W/2),E,W)
960 CALL SOUND(100,110,0)
970 FOR I=0 TO 30 STEP 5
980 CALL SOUND(-40,110,I,118,I,121,I,-5,I)
990 NEXT I
1000 REM KONTROLLE SPIELENDE
1010 CALL GCHAR(18,25,U)
1020 IF U=65 THEN 1870
1030 IF C=65 THEN 2350
1040 IF C=70 THEN 2390
1050 IF C=78 THEN 2430
1060 IF C=80 THEN 1830
1070 C=C+1
1080 GOTO 490
1090 REM PRINTROUTINE
1100 FOR P=1 TO LEN(T$)
1110 CALL HCHAR(24,3+P,ASC(SEG$(T$,P,1)))
1120 NEXT P
1130 CALL SOUND(40,T,0)
1140 RETURN
1150 REM ALTER STAPEL 1
1160 JA=1
1170 JE=6
1180 S=15
1190 RETURN
1200 REM ALTER STAPEL 2
1210 JA=8
1220 JE=13
1230 S=8
1240 RETURN
1250 REM ALTER STAPEL 3
1260 JA=18
1270 JE=23
1280 S=25
1290 RETURN
1300 REM NEUER STAPEL 1
1310 ZA=1
1320 ZE=6
1330 SP=15
1340 RETURN
1350 REM NEUER STAPEL 2
1360 ZA=8
1370 ZE=13
1380 SP=8
1390 RETURN
1400 REM NEUER STAPEL 3
1410 ZA=18
1420 ZE=23
1430 SP=25
1440 RETURN
1450 REM  ANZEIGE TAGE
1460 P$=C$&". tag"
1470 FOR P=1 TO LEN(P$)
1480 CALL HCHAR(23,3+P,ASC(SEG$(P$,P,1)))
1490 NEXT P
1500 RETURN
1510 REM  TITEL UND SPIEL-{12*(SPACE)}REGELN
1520 CALL CLEAR
1530 PRINT "{28*(*)}"
1540 PRINT "*{26*(SPACE)}*"
1550 PRINT "*     p y r a m i d e{6*(SPACE)}*"
```

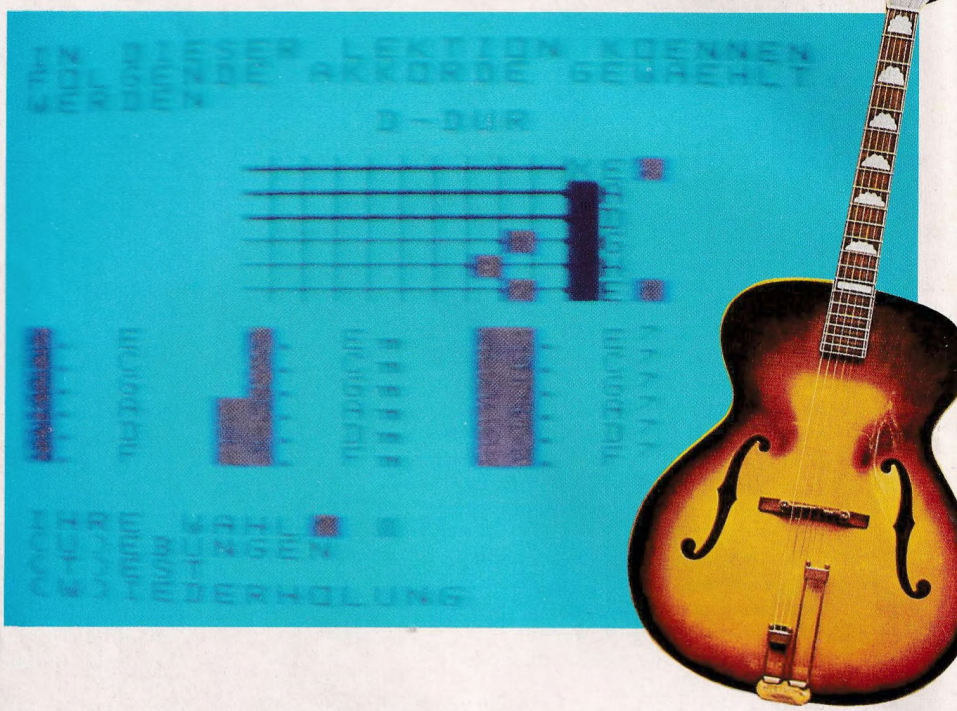


```
1560 PRINT "*{26*(SPACE)}*"
1570 PRINT "{28*(*)}": : : : : : :
1580 PRINT "benoetigen sie die{10*(SPACE)}
     spielregeln?          <j/n>"
1590 CALL KEY(0,KEY,ST)
1600 IF ST=0 THEN 1590
1610 IF KEY=78 THEN 1620 ELSE 1630
1620 RETURN
1630 CALL CLEAR
1640 PRINT "der pharao will umziehen!
       als hervorragender bau-
       meister aegyptens werden sie"
1650 PRINT "damit beauftragt, die in
       sechs bauteile zerlegte
       pyramide von ihrem ur-"
1660 PRINT "spruenglichen standort (1)
       zu ihrem neuen standort (3)
      zu transportieren, wo der"
1670 PRINT "pharao seinen reichtuemern  naeher ist. "
1680 PRINT "da aus statischen gruenden
       niemals ein groesseres bau- teil auf einem
      kleineren "
1690 PRINT "gelagert werden kann, be-
       noetigen sie ein zwischen-  lager (2)
     . zur bewaeltigung "
1700 PRINT "ihrer aufgabe haben sie
       genau 80 tage zeit. jeden
       tag kann nur ein bauteil"
1710 PRINT "transportiert werden.": :
1720 PRINT "bitte eine taste druecken!"
1730 CALL KEY(0,KEY,ST)
1740 IF ST=0 THEN 1730
1750 CALL CLEAR
1760 PRINT "sollten sie versagen, werdensie nach
      ablauf der frist   den heiligen nil-krokodilen
      "
1770 PRINT "zum fruehstueck serviert.
       haben sie jedoch erfolg,    werden sie in gold
     {10*(SPACE)}aufgewogen.": : : :
1780 PRINT "{7*(SPACE)}viel erfolg!{9*(SPACE)}": : :
      : :
1790 PRINT "bitte eine taste druecken!"
1800 CALL KEY(0,KEY,ST)
1810 IF ST=0 THEN 1800
1820 RETURN
1830 PRINT "sie haben versagt!": :
1840 PRINT "obwohl selbst die heiligen
       nil-krokodile sie ver-{6*(SPACE)}
     schmaehten, gibt ihnen der"
1850 PRINT "pharao gelegenheit, ihr
       koennen nochmals zu beweisen.": :
1860 GOTO 1910
1870 PRINT "herzlichen glueckwunsch!
       sie haben ihre aufgabe"
1880 PRINT "geloest und werden wie{6*(SPACE)}
     versprochen belohnt.": :
1890 PRINT "aufgrund ihres ausschwei-
       fenden lebenswandels sind"
1900 PRINT "sie aber nach kurzer zeit
       gezwungen, wieder geld zu   verdienen.": :
1910 PRINT "druecken sie eine taste,
       wenn sie bereit sind."
1920 PRINT "druecken sie <space>, wenn
       sie sich der herausforderungnicht stellen
      wollen."
1930 CALL KEY(0,KEY,ST)
1940 IF ST=0 THEN 1930
1950 IF KEY<>32 THEN 410
1960 CALL VCHAR(1,1,32,768)
1970 END
1980 REM GRAFIK
1990 FOR I=1 TO 124
2000 READ Q(I),X(I)
2010 CALL HCHAR(Q(I),X(I),80)
2020 NEXT I
2030 DATA 23,4,23,5,22,5,23,6,22,6,21,6,21,7,22,7,23,
     7,20,8,21,8,22,8,23,8,19,9,20,9
2040 DATA 21,9,22,9,23,9,18,10,19,10,20,10,21,10,22,
     10,18,11,19,11,20,11,21,11,22,11
2050 DATA 18,12,19,12,20,12,21,12,17,13,18,13,19,13,
     20,13,17,14,18,14,19,14,20,14
2060 DATA 16,15,17,15,18,15,19,15,16,16,17,16,18,16,
     19,16,15,17,16,17,17,17,18,17
2070 DATA 15,18,16,18,17,18,18,18,15,19,16,19,17,19,
     18,19,15,20,16,20,17,20,18,20
2080 DATA 15,21,16,21,17,21,14,22,15,22,16,22,17,22,
     13,23,14,23,15,23,16,23,12,24
2090 DATA 13,24,14,24,15,24,16,24,11,25,12,25,13,25,
     14,25,15,25,11,26,12,26,13,26
2100 DATA 14,26,15,26,11,27,12,27,13,27,14,27,10,28,
     11,28,12,28,13,28,14,28
2110 DATA 9,29,10,29,11,29,12,29,13,29,14,29,8,30,9,
     30,10,30,11,30,12,30,13,30
2120 DATA 7,31,8,31,9,31,10,31,11,31,12,31,6,32,7,32,
     8,32,9,32,10,32,11,32,12,32
2130 FOR I=1 TO 24
2140 READ Q(I),X(I),Y(I)
2150 CALL HCHAR(Q(I),X(I),Y(I))
2160 NEXT I
2170 DATA 6,22,88,5,22,89,4,21,93,4,22,90,4,23,95,3,
     21,92,3,22,91,3,23,94
2180 DATA 4,25,88,3,25,89,2,24,93,2,25,90,2,26,95,1,
     24,92,1,25,91,1,26,94
2190 DATA 5,7,88,4,7,89,3,6,93,3,7,90,3,8,95,2,6,92,
     2,7,91,2,8,94
2200 FOR I=1 TO 24
2210 READ Q(I),X(I),Y(I)
2220 CALL HCHAR(Q(I),X(I),Y(I))
2230 NEXT I
2240 DATA 23,16,78,23,17,79,22,16,76,22,17,77,21,16,
     74,21,17,75,20,16,72,20,17,73
2250 DATA 18,31,78,18,32,79,17,31,76,17,32,77,16,31,
     74,16,32,75,15,31,72,15,32,73
2260 DATA 22,18,78,22,19,79,21,18,76,21,19,77,20,18,
     74,20,19,75,19,18,72,19,19,73
2270 RESTORE
2280 CALL HCHAR(1,15,65)
2290 CALL HCHAR(2,14,66,3)
2300 CALL HCHAR(3,13,67,5)
2310 CALL HCHAR(4,12,68,7)
2320 CALL HCHAR(5,11,69,9)
2330 CALL HCHAR(6,10,70,11)
2340 RETURN
2350 CALL HCHAR(17,9,35)
2360 CALL HCHAR(17,10,36)
2370 CALL HCHAR(17,11,37)
2380 GOTO 1060
2390 CALL HCHAR(14,18,35)
2400 CALL HCHAR(14,19,36)
2410 CALL HCHAR(14,20,37)
2420 GOTO 1060
2430 CALL HCHAR(17,23,35)
2440 CALL HCHAR(17,24,36)
2450 CALL HCHAR(17,25,37)
2460 GOTO 1060
```



Saiten-Blick

Mit der Gitarre auf Du und Du in EX BASIC



Manfred Kiwitt aus Hamburg hat uns ein Programm zugeschickt, das den Gitarre-interessierten User ansprechen soll. Mit diesem Programm will Manfred das Erlernen und Üben der vielen verschiedenen Griffe und Akkorde erleichtern. Selbst blutige Anfänger, so wie wir hier in der Redaktion, können mit diesem Listing das muntere Klimpern erlernen und üben, denn ein detailliertes "HOW TO DO" ist in das Programm integriert. Darum hier nur kurz der Aufbau:

Zeile 100-360 : Initialisierung, Datensätze
Zeile 370-490 : Gitarre stimmen
Zeile 500-1040: Anweisungen
Zeile 1050-1110: Menü
Zeile 1120-1290: Test/Wiederholung
Zeile 1300-1470: Wahl des Datensatzes
Zeile 1480-1520: Übungen

```
100 T10=0
110 RANDOMIZE
120 ON WARNING NEXT
130 DATA 165,220,294,392,494,659
140 CALL COLOR(3,16,14,4,16,14)
150 DATA E6,A,D,G,H,E1
160 DATA E-DUR,9,17,7,16,8,16,12,32,12,
    32,12,32,165,247,330,415,494,659,
    165,208,247
170 DATA C-DUR,10,17,8,16,6,15,7,15,12,
    32,12,32,196,262,330,392,523,659,
    262,330,392
180 DATA G-DUR,7,16,6,15,11,15,12,32,
    12,32,12,32,196,247,294,392,494,
    784,196,247,294
190 DATA A-DUR,8,16,9,16,10,16,12,32,
    12,32,12,32,165,220,330,440,554,
    659,220,277,330
200 DATA D-DUR,9,16,11,16,10,15,12,32,
    6,18,12,32,110,220,294,440,587,740,
    294,370,440
210 DATA F-DUR,5,17,9,16,7,15,8,15,12,
    32,12,32,175,262,349,440,523,698,
    175,220,262
220 DATA E-MOLL,12,32,7,16,8,16,12,32,
    12,32,12,32,165,247,330,392,494,
    659,165,196,247
230 DATA C-MOLL,5,15,10,14,8,13,9,13,
    12,32,12,32,196,262,392,523,622,
    784,262,311,392
240 DATA G-MOLL,5,15,7,13,8,13,12,32,
    12,32,12,32,196,294,392,466,587,
    784,196,233,294
250 DATA A-MOLL,10,17,8,16,9,16,12,32,
    12,32,12,32,165,220,330,440,523,
    659,220,262,330
260 DATA D-MOLL,11,17,9,16,10,15,12,32,
    6,18,12,32,-3,220,294,440,587,698,
    294,349,440
270 DATA F-MOLL,5,17,12,32,7,15,8,15,
    12,32,12,32,175,262,349,415,523,
    698,175,208,262
280 DATA Eh,9,17,7,16,8,16,10,15,12,32,
    12,32,165,247,330,415,587,659,247,
    311,277
290 DATA Ch,10,17,8,16,7,15,9,15,12,
    32,12,32,165,262,330,466,523,659,
    262,311,233
300 DATA Gh,11,17,7,16,6,15,12,32,12,
    32,12,32,196,247,294,392,494,698,
    196,247,349
310 DATA Ah,8,16,12,32,10,16,12,32,12,
    32,12,32,165,220,330,392,554,659,
    311,196,294
320 DATA Dh,10,17,9,16,11,16,12,32,6,
    18,12,32,-3,220,294,440,523,740,
    220,262,370
330 DATA Fh,5,17,9,16,7,15,12,32,12,32,
    12,32,175,262,311,440,523,698,262,
    294,220
340 DISPLAY AT(12,8)ERASE ALL:
    "GITARREN-KURS"
350 CALL CHAR(100,"{16*(F)}",
    101,"010101FFFF010101",102,
    "010101FF01010101")::
     CALL CHAR(103,"{8*(0)}
    7E2A2A2A",104,"3C04081020")
360 FOR Z=1 TO 6 :: READ X(Z)::
     NEXT Z :: FOR B=1 TO 6 ::
```

```
     READ A$(B):: NEXT B
370 DISPLAY AT(7,1)ERASE ALL:
    "STIMMEN SIE ZUERST": :
    "IHRE GITARRE": :
    "WENN SIE BEREIT SIND,": :
    "DRUECKEN SIE  {CHR$(123)}
    ENTER{CHR$(125)}"
380 CALL KEY(0,R,S):: IF R<>13 THEN 380
390 Z=1 :: CALL CLEAR
400 DISPLAY AT(14,1):
    "DIE";Z;".";"{CHR$(123)}
    ";A$(Z);"{CHR$(125)}-SAITE":
    "MUSS SO KLINGEN!":""
410 DISPLAY AT(17,1):
    "{CHR$(123)}P{CHR$(125)}{CHR$(126)}
    PAUSE": :"{CHR$(123)}
    SPACE{CHR$(125)}{CHR$(126)}
    PAUSE ENDE": :"{CHR$(123)}
    ENTER{CHR$(125)}{CHR$(126)}
    NAECHSTER TON"
420 CALL SOUND(-300,X(Z),0)::
     CALL KEY(0,R,S):: IF R=80
    THEN 1280 ELSE IF R<>13 THEN 420
430 CALL SOUND(1,1000,30):: Z=Z+1 ::
     IF Z<=6 THEN 400
440 DISPLAY AT(22,1):
    "{CHR$(123)}W{CHR$(125)}
    IEDERHOLUNG": :"{10*(SPACE)}
    WEITER MIT {CHR$(123)}
    ENTER{CHR$(125)}"
450 CALL KEY(0,R,S):: IF R=87
    THEN 390 ELSE IF R<>13 THEN 450
460 DISPLAY AT(9,1)ERASE ALL:
    "JETZT SOLLTE IHRE GITARRE": :
    "GESTIMMT SEIN.": : :
    "VERGLEICHEN SIE VON ": :
    "OBEN NACH UNTEN."
470 FOR I=1 TO 6 :: CALL SOUND(500,X(I)
    ,0):: NEXT I
480 DISPLAY AT(18,1):
    "{CHR$(123)}W{CHR$(125)}
    IEDERHOLUNG": :"{CHR$(123)}
    S{CHR$(125)} NEU STIMMEN": : : :
    "{10*(SPACE)}WEITER MIT {CHR$(123)}
    ENTER{CHR$(125)}"
490 CALL KEY(0,R,S):: IF R=87
    THEN 470 ELSE IF R=83
    THEN 390 ELSE IF R<>13 THEN 490
500 DISPLAY AT(12,1)ERASE ALL:
    "FANGEN WIR ALSO AN!": :
    "BRAUCHEN SIE EINE": :
    " ERKLAERUNG?": :
    "1  {CHR$(126)} JA": :
    "2  {CHR$(126)} NEIN"
510 CALL KEY(0,R,S):: IF R=49
    THEN 530 ELSE IF R<>50 THEN 510
520 CALL CLEAR :: GOTO 1050
530 DISPLAY AT(9,1)ERASE ALL:
    "WENN SIE IHREN COMPUTER MAL ": :
    "NICHT BEI SICH HABEN,": :
    "SOLLTEN SIE EINE": :
    "'STIMMFLOETE'"
540 DISPLAY AT(17,1):
    "BEI SICH TRAGEN.": :
    "SIE GIBT GENAU DIE GLEICHEN": :
    "TOENE VON SICH.": : :
    "{10*(SPACE)}WEITER MIT {CHR$(123)}
    ENTER{CHR$(125)}"
550 CALL KEY(0,R,S):: IF R<>13 THEN 550
560 DISPLAY AT(1,1)ERASE ALL:
    "HABEN SIE KEINES VON BEIDEM,":
    "MUESSEN SIE SICH EINEN TON":
    "GEBEN LASSEN.": :
    "NEHMEN WIR AN,ES IST DAS
    {CHR$(123)}E{CHR$(125)}"
570 DISPLAY AT(6,1):
    "STIMMEN SIE DIE SAITE EIN.": :
    "DANN DRUECKEN SIE DIE ":
    "SAITE {CHR$(123)}E{CHR$(125)}":
    "NACH DEM 5.BUNDSTAEBCHEN UND":
    "STIMMEN DIE NAECHSTE,{CHR$(123)}
    A{CHR$(125)}-SAITE,NACH DIESEM TON"
580 DISPLAY AT(13,1):
    "ALLE SAITEN,WERDEN SO GESTIMMT.":
     :"NUR DIE 5.{CHR$(123)}
    H{CHR$(125)} SAITE": :
    "WIRD NACH DER 4.{CHR$(123)}
    G{CHR$(125)} SAITE": :
```


HAASE-Computersysteme – Ihr ATARI-Fachmann:

CPM für Atari
Rana-Floppy
Touch-Tablett
Erweiterungsber.

dt. Literatur

viele Adventure und
Actiongarnes

Basic XL
Compiler

Telefonmodem
Drucker

Flightsimulator, Dimension X, Moon Patrol, Dark Crystal, N. Y. C.

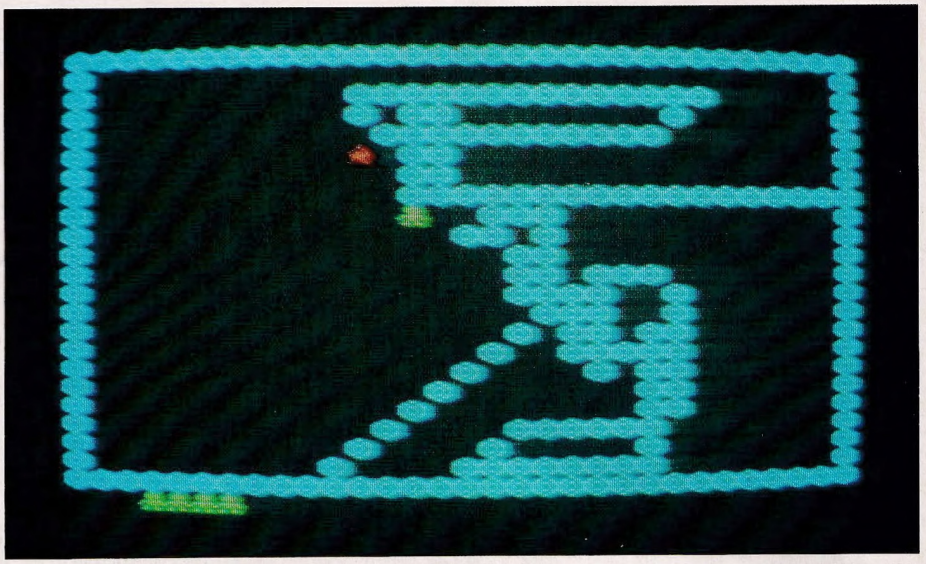
und das alles bei **HAASE-Computersysteme,** Wiedfeldstr. 11, 4300 Essen 1
Info anfordern: Telefon 02 01 – 42 25 75

```
    "IM 4.BUND GESTIMMT."
590 DISPLAY AT(21,1):"":"":
    "{10*(SPACE)}WEITER MIT {CHR$(123)}
    ENTER{CHR$(125)}"
600 CALL KEY(0,R,S):: IF R<>13 THEN 600
610 DISPLAY AT(1,1)ERASE ALL:
    "DAS GRIFFBRETT":
    "WIRD WIE FOLGT DARGESTELLT" ::
     GOSUB 620 :: GOTO 630
620 CALL VCHAR(6,20,100,6)::
     FOR I=1 TO 3 ::
     CALL HCHAR(I+5,10,101,10)::
     CALL HCHAR(I+8,10,102,10)::
     NEXT I :: CALL HCHAR(5,1,32,32)::
     RETURN
630 DISPLAY AT(13,1):"WOBEI":
    "DER SENKRECHTE DICKE STRICH DEN
     SATTEL ":"DIE DUENNEN
     SENKRECHTEN":"STRICHE DIE BUNDE
     UND"
640 DISPLAY AT(18,1):
    "DIE WAAGERECHTEN STRICHE":
    "DIE SAITEN": :"DARSTELLEN.": : :
    "{10*(SPACE)}WEITER MIT {CHR$(123)}
    ENTER{CHR$(125)}"
650 CALL KEY(0,R,S):: IF R<>13 THEN 650
660 DISPLAY AT(1,1):"DIE SAITEN":
    "WERDEN WIE FOLGT BEZEICHNET"
670 FOR I=1 TO 6 :: DISPLAY AT(I+5,19):
    A$(I):: NEXT I
680 DISPLAY AT(14,1)
690 DISPLAY AT(15,1):
    "{CHR$(123)}E6{CHR$(125)}
     DIE DICKSTE": :"UND": :
    "{CHR$(123)}E1{CHR$(125)}
     DIE DUENNSTE": :"SAITE IST."
700 CALL KEY(0,R,S):: IF R<>13 THEN 700
710 DISPLAY AT(1,1):
    "EIN AKKORD  {CHR$(123)}
     GRIFF {CHR$(125)}":
    "SIEHT FOLGENDERMASSEN AUS"
720 RESTORE 160 :: GOSUB 730 ::
     GOTO 770
730 READ B$,A,P,C,D,E,F,G,H,A1,B1,C1,
    D1,J,K,L,M,N,O,A2,B2,C2
740 DISPLAY AT(4,12):
    B$ :: DISPLAY AT(A,P)SIZE(1):
    "1" :: DISPLAY AT(C,D)SIZE(1):
    "2" :: DISPLAY AT(E,F)SIZE(1):
    "3" :: DISPLAY AT(G,H)SIZE(1):"4"
750 DISPLAY AT(A1,B1)SIZE(1):
    "X" :: DISPLAY AT(C1,D1)SIZE(1):"X"
760 CALL HCHAR(12,6,32)::
     CALL HCHAR(5,19,32):: RETURN
770 DISPLAY AT(15,1):
    "1 DEN ZEIGEFINGER": :
    "2 DEN MITTELFINGER": :
    "3 DEN RINGFINGER": :
    "4 DEN KLEINEN FINGER": :"ZEIGEN."
780 CALL KEY(0,R,S):: IF R<>13 THEN 780
790 DISPLAY AT(13,1):
    "DRUECKEN SIE MIT DEM AN-": :
    "GEZEIGTEN FINGER DIE SAITE": :
    "HINTER DEM BUNDSTAEBCHEN": :
    "SO NIEDER,DASS EIN REINER": :
    "TON ENTSTEHT.BERUEHREN"
800 DISPLAY AT(23,1):
    "SIE KEINE ANDERE SAITE."
810 CALL KEY(0,R,S):: IF R<>13 THEN 810
820 RESTORE 210 :: GOSUB 620 ::
     GOSUB 730
830 CALL VCHAR(A+1,P+2,49,6)
840 DISPLAY AT(13,1):
    "6 X DIE 1 AUF DEM GRIFFBRETT": :
    "ZEIGT EINEN 'BARRE'-GRIFF AN": :
    "DAS HEISST,DER BEZEICHNETE": :
    "FINGER DRUECKT {CHR$(123)}
    ALLE 6{CHR$(125)}"
850 DISPLAY AT(20,1):"":
    "SAITEN IN DEM BUND.":"":""
860 CALL KEY(0,R,S):: IF R<>13 THEN 860
870 RESTORE 260 :: GOSUB 620 ::
     GOSUB 730
880 DISPLAY AT(13,1):
    "EIN 'X' AUF DEM SATTEL":
    "BEDEUTET,DASS DIESE SAITE":
    "'STUMM' GEMACHT WIRD.":
    "DIE SAITE WIRD MIT DEM AM"
890 DISPLAY AT(17,1):
    "NAECHSTEN LIGENDEN FINGER":
    "SO BERUEHRT,DASS SIE ":
    "KEINEN VOLLEN TON VON SICH":
    "GIBT.":"DIE SAITE {CHR$(123)}
    E6{CHR$(125)} SOLLTE MIT"
900 DISPLAY AT(22,1):
    "DEM DAUMEN 'ABGEDAEMPFT'":
    "WERDEN."
910 CALL KEY(0,R,S):: IF R<>13 THEN 910
920 DISPLAY AT(1,1):"": :
     :: DISPLAY AT(13,1):"": : : :
    "BIS HIER ALLES KLAR?": :
    "DRUECKEN SIE": :
    "{CHR$(123)}W{CHR$(125)}
     FUER WIEDERHOLUNG": : :
930 CALL KEY(0,R,S):: IF R=87
    THEN 530 ELSE IF R<>13 THEN 930
940 DISPLAY AT(13,1):
    "GLEICHZEITIG WIRD DER": :
    "AKKORD ABGESPIELT.": :"UND ZWAR":
     :"1. JEDE SAITE": :
    "2. EIN AKKORDAEHNLICHER TON":"":
    "":""
950 Q=500
960 CALL SOUND(Q,J,0)::
     CALL SOUND(Q,K,0)::
     CALL SOUND(Q,L,0)::
     CALL SOUND(Q,M,0)::
     CALL SOUND(Q,N,0)::
     CALL SOUND(Q,O,0)
970 FOR Y=1 TO 1000 :: NEXT Y ::
     FOR Y1=1 TO 30 ::
     CALL SOUND(-150,A2,Y1,B2,Y1,C2,Y1)
    :: FOR Y2=1 TO 15 :: NEXT Y2 ::
     NEXT Y1
980 DISPLAY AT(24,10):
    "WEITER MIT {CHR$(123)}
    ENTER{CHR$(125)}"
990 T10=T10+1 :: IF T10>1 THEN 1010
1000 CALL KEY(0,R,S):: IF R<>13
    THEN 1000
1010 DISPLAY AT(13,1):
    "SIE KOENNEN DEN AKKORD BE-": :
    "LIEBIG OFT HOEREN.": :
    "DRUECKEN SIE NUR ": :
    "{CHR$(123)}W{CHR$(125)}": :
    "FUER {CHR$(123)}W{CHR$(125)}
    IEDERHOLUNG.":"":""
1020 CALL KEY(0,R,S):: IF R=87
```

```
    THEN 950 ELSE IF R<>13 THEN 1020
1030 T10=0 :: DISPLAY AT(13,1):"": :
    "ALLES KLAR?": :
    "WENN SIE ALLES NOCH EINMAL": :
    "WIEDERHOLEN MOECHTEN,": :
    "DRUECKEN SIE": :
    "{CHR$(123)}W{CHR$(125)}
    IEDERHOLUNG"
1040 CALL KEY(0,R,S):: IF R=87
    THEN 530 ELSE IF R<>13 THEN 1040
1050 DISPLAY AT(1,1):
    "IN DIESER LEKTION KOENNEN":
    "FOLGENDE AKKORDE GEWAEHLT":
    "WERDEN"
1060 DISPLAY AT(13,1):
    "1. E   7. Eg  13. Eh":
    "2. C   8. Cg  14. Ch":
    "3. G   9. Gg  15. Gh":
    "4. A  10. Ag  16. Ah":
    "5. D  11. Dg  17. Dh"
1070 DISPLAY AT(18,1):
    "6. F  12. Fg  18. Fh":"":"":""
1080 GOSUB 620
1090 FOR I=1 TO 6 :: DISPLAY AT(I+5,19):
    A$(I):: NEXT I
1100 T1=50 :: DISPLAY AT(21,1):
    "IHRE WAHL?":"{CHR$(123)}
    U{CHR$(125)}EBUNGEN":
    "{CHR$(123)}T{CHR$(125)}EST":
    "{CHR$(123)}W{CHR$(125)}
    IEDERHOLUNG"
1110 ACCEPT AT(21,12)
    VALIDATE(DIGIT,"WUT"):
    R$ :: IF R$="W"
    THEN 1230 ELSE IF R$="U"
    THEN GOSUB 1480 ELSE IF R$="T"
    THEN V=1
1120 IF T>0 OR V=1 THEN R=INT(RND*18)
    +1 ELSE 1140
1130 R$=STR$(R):: DISPLAY AT(22,1):
    "NOCH";T;"AKKORDE": :
    "MIT GESCHWINDIGKEIT";
    T21140 IF VAL(R$)<1 OR VAL(R$)>18
    THEN 1110
1150 ON VAL(R$)GOSUB 1300,1310,1320,
    1330,1340,1350,1360,1370,1380,1390,
    1400,1410,1420,1430,1440,1450,1460,
    1470
1160 READ B$,A,P,C,D,E,F,G,H,A1,B1,C1,
    D1,J,K,L,M,N,O,A2,B2,C2
1170 GOSUB 620
1180 IF V=1 THEN GOSUB 1490
1190 IF A=5 THEN CALL
     VCHAR(A+1,P+2,49,6):: A=A+1
1200 DISPLAY AT(4,12):
    B$ :: DISPLAY AT(A,P)SIZE(1):
    "1" :: DISPLAY AT(C,D)SIZE(1):
    "2" :: DISPLAY AT(E,F)SIZE(1):
    "3" :: DISPLAY AT(G,H)SIZE(1):"4"
1210 DISPLAY AT(A1,B1)SIZE(1):
    "X" :: DISPLAY AT(C1,D1)SIZE(1):"X"
1220 CALL HCHAR(12,6,32)
1230 Q=500
1240 CALL SOUND(Q,J,0)::
     CALL SOUND(Q,K,0)::
     CALL SOUND(Q,L,0)::
     CALL SOUND(Q,M,0)::
     CALL SOUND(Q,N,0)::
     CALL SOUND(Q,O,0)
1250 FOR Y=1 TO 300 :: NEXT Y ::
     FOR Y1=1 TO 30 ::
     CALL SOUND(-150,A2,Y1,B2,Y1,C2,Y1)
    :: FOR Y2=1 TO 15 :: NEXT Y2 ::
     NEXT Y1
1260 FOR I=1 TO T1 :: NEXT I
1270 T=T-1 :: IF T>0 OR V=1
    THEN 1120 ELSE 1100
1280 CALL KEY(0,R1,S1):: IF R1=32
    THEN 420 ELSE IF R1=13
    THEN 430 :: GOTO 1280
1290 END
1300 RESTORE 160 :: RETURN
1310 RESTORE 170 :: RETURN
1320 RESTORE 180 :: RETURN
1330 RESTORE 190 :: RETURN
1340 RESTORE 200 :: RETURN
1350 RESTORE 210 :: RETURN
1360 RESTORE 220 :: RETURN
1370 RESTORE 230 :: RETURN
1380 RESTORE 240 :: RETURN
1390 RESTORE 250 :: RETURN
1400 RESTORE 260 :: RETURN
1410 RESTORE 270 :: RETURN
1420 RESTORE 280 :: RETURN
1430 RESTORE 290 :: RETURN
1440 RESTORE 300 :: RETURN
1450 RESTORE 310 :: RETURN
1460 RESTORE 320 :: RETURN
1470 RESTORE 330 :: RETURN
1480 DISPLAY AT(22,1):
    "WIVIELE AKKORDE?":
    "GESCHWINDIGKEIT?":
    "<1{CHR$(126)}SCHNELL>" ::
     ACCEPT AT(22,18):
    T :: ACCEPT AT(24,18):
    T2 :: T1=T2*100 :: RETURN
1490 V1=V1+1 :: DISPLAY AT(4,12):
    B$ :: DISPLAY AT(21,1):
    "FUER LOESUNG {CHR$(123)}
    ENTER{CHR$(125)}
    " :: DISPLAY AT(22,5)SIZE(-3):
    11-V1 :: DISPLAY AT(24,1):
    "OHNE WIEDERHOLUNG"
1500 CALL KEY(0,R,S):: IF R<>13
    THEN 1500
1510 IF V1=10 THEN V=0 :: T=0 :: V1=0
1520 RETURN
```

ATARI – GENIE 16 – EPSON – STAR
Überraschungs-Preisliste anfordern!
TEL.: 02623-6676
COMPY SHOP
DER ATARI-SPEZIALIST
5433
SIERSHAHN
BERGSTR. 18

TEUFEL



Höllische Vergnügen,
programmiert von
Christoph Bladoschewski
in BASIC

Bei diesem Spiel geht es darum, kleine rote Teufelchen durch Einkreisen zu vernichten. Auf jedem Feld, das ihr einmal belegt habt, entsteht ein Symbol, das die Teufel vernichtet, sobald sie damit in Berührung kommen. Für den Spieler sind diese Felder ungefährlich; man kann sich sogar hinter den Symbolen verstecken, wenn man vor einem Satan flüchten muß.

Am Anfang des Spiels ist das noch relativ einfach, denn man hat es nur mit einem Gegner zu tun. Nach und nach steigert sich der Schwierigkeitsgrad des Spiels auf bis zu 10 Höllenknechte.

Die Steuerung erfolgt mit Joystick 1. Punkte und Highscore werden am Schluß des Spiels angezeigt. Im übrigen solltet ihr nicht verzagen, wenn ihr bereits in der ersten Runde draufgeht; mit der richtigen Strategie sind schnell hohe Highscores zu erreichen.

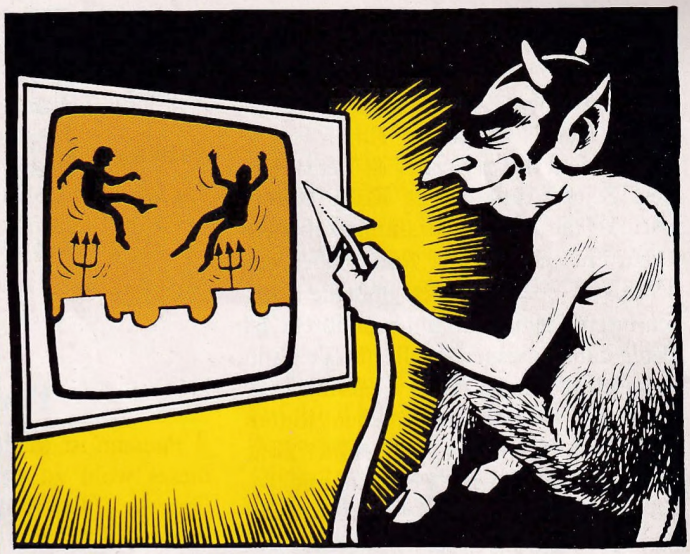
Programmstruktur:

Zeile
- 240- 300 Titelbild
- 340- 770 Programmvorbereitung (Zeichendefinition, Variablenzuweisung, Graphikaufbau)
- 810-1160 Hauptprogramm
- 1200-1340 Gefangennahme eines Teufels
- 1380-1560 Ende und Statistik

AUF TEUFEL KOMM RAUS

```
100 REM{20*(*)}
110 REM*{18*(SPACE)}*
120 REM*{6*(SPACE)}TEUFEL{6*(SPACE)}*
130 REM*{18*(SPACE)}*
140 REM{20*(*)}
150 REM*  ENTWICKELT VON  *
160 REM*{18*(SPACE)}*
170 REM* CH.BLADOSCHEWSKI *
180 REM*{18*(SPACE)}*
190 REM*{11*(SPACE)}(C)1983*
200 REM{20*(*)}
210 REM
220 REM TITELBILD
230 REM
240 CALL CLEAR
250 CALL SCREEN(8)
260 PRINT TAB(8);"T E U F E L": : : : : : : : : : :
    :
270 CALL CHAR(96,"3C4299A1A199423C")
280 PRINT "{CHR$(96)} 1983 BY CH. BLADOSCHEWSKI"
290 FOR I=1 TO 1000
300 NEXT I
310 REM
320 REM VORBEREITUNG
330 REM
340 CALL CLEAR
350 SC=0
360 RU=0
370 UM=0
380 LEB=5
390 CALL SCREEN(2)
400 RANDOMIZE
410 CALL CHAR(96,"183C3C187E992466")
420 CALL CHAR(104,"1866DBA5A5DB6618")
430 CALL CHAR(112,"0818245A7E3C2442")
440 CALL COLOR(3,16,2)
450 CALL COLOR(4,16,2)
460 CALL COLOR(9,4,1)
470 CALL COLOR(10,8,1)
480 CALL COLOR(11,9,1)
490 RU=RU+1
500 FOR I=2 TO 22
510 CALL HCHAR(I,3,32,29)
520 NEXT I
530 CALL HCHAR(1,2,104,31)
540 CALL HCHAR(23,2,104,31)
550 CALL VCHAR(2,2,104,21)
560 CALL VCHAR(2,32,104,21)
570 CALL HCHAR(24,5,96,LEB)
580 IF RU<11 THEN 600
590 RU=10
600 UM=0
610 FOR I=110 TO 500 STEP 30
620 CALL SOUND(20,I,0)
630 NEXT I
640 FOR I=1 TO RU
650 X=21*RND+2
660 Y=29*RND+3
670 CALL GCHAR(X,Y,T)
680 IF T<>32 THEN 650
690 GEA(I)=X
700 GEB(I)=Y
```

```
710 CALL HCHAR(X,Y,112)
720 NEXT I
730 A=21*RND+2
740 B=29*RND+2
750 CALL GCHAR(A,B,T)
760 IF T<>32 THEN 730
770 CALL HCHAR(A,B,96)
780 REM
790 REM HAUPTPROGRAMM
800 REM
810 FOR I=1 TO RU
820 IF GEA(I)=0 THEN 970
830 L=A-GEA(I)
840 M=B-GEB(I)
850 CALL HCHAR(GEA(I),GEB(I),32)
860 CALL GCHAR(GEA(I)+SGN(L),GEB(I)+SGN(M),T)
870 IF T=104 THEN 910
880 GEA(I)=GEA(I)+SGN(L)
890 GEB(I)=GEB(I)+SGN(M)
900 GOTO 930
910 GEA(I)=GEA(I)-SGN(L)
920 GEB(I)=GEB(I)-SGN(M)
930 CALL GCHAR(GEA(I),GEB(I),T)
940 IF T=96 THEN 1040
950 IF T<>32 THEN 1200
960 CALL HCHAR(GEA(I),GEB(I),112)
970 CALL JOYST(1,Y,X)
980 CALL HCHAR(A,B,104)
990 A=A-SGN(X)
1000 B=B+SGN(Y)
1010 IF (A<1)+(A>23)+(B<2)+(B>32)<0 THEN 1400
1020 CALL GCHAR(A,B,T)
1030 IF T<>112 THEN 1130
1040 LEB=LEB-1
1050 CALL SOUND(-500,2200,0,2204,3,2116,1,-6,0)
1060 IF LEB=0 THEN 1380
1070 A=20*RND+2
1080 B=28*RND+3
1090 CALL GCHAR(A,B,T)
1100 IF T<>32 THEN 1070
1110 CALL HCHAR(A,B,96)
1120 CALL HCHAR(24,LEB+5,32)
1130 CALL SOUND(-200,-4,0)
1140 CALL HCHAR(A,B,96)
1150 NEXT I
1160 GOTO 810
1170 REM
1180 REM TEUFEL TOT
1190 REM
1200 GEA(I)=0
1210 CALL SOUND(100,247,0)
1220 CALL SOUND(200,362,0)
1230 CALL SOUND(100,349,0)
1240 CALL SOUND(100,330,0)
1250 UM=UM+1
1260 SC=SC+UM*RU+LEB*100
1270 CALL HCHAR(24,15,32,6)
1280 PUN=SC
1290 FOR Q=5 TO 0 STEP -1
1300 PUT=INT(PUN/10^Q)
1310 PUN=PUN-PUT*10^Q
1320 CALL HCHAR(24,20-Q,PUT+48)
1330 NEXT Q
1340 IF RU=UM THEN 490 ELSE 1150
1350 REM
1360 REM ENDE
1370 REM
```



```
1380 CALL SOUND(200,-7,0)
1390 CALL HCHAR(A,B,96)
1400 CALL CHAR(96,"1818FFFF18181818")
1410 FOR I=1000 TO 110 STEP -20
1420 CALL SOUND(20,I,0)
1430 NEXT I
1440 CALL CLEAR
1450 CALL SCREEN(8)
1460 PRINT TAB(10);"GAME OVER!": : : : : : : : : :
1470 PRINT "YOUR SCORE:";SC: : : : : :
1480 IF SC<=HS THEN 1530
1490 FOR I=110 TO 550 STEP 20
1500 CALL SOUND(20,I,0)
1510 NEXT I
1520 HS=SC
1530 PRINT "HIGHSCORE :";HS: : : : : :
1540 PRINT "NEUES SPIEL = FEUERKNOPF"
1550 CALL KEY(1,K,S)
1560 IF S=0 THEN 1550 ELSE 340 ■
```


ACHTUNG!
Bei uns finden Sie fast alles für Ihren
VC-20/64

Hardware:
Speichererweiterungen 8K-64K RAM, Moduladapter 2/3/5-fach, Recorderinterface, Schnell-Save Module, Toolkitmodule, 40/80 Zeichenkarten, Epromprogrammierer, Epromkarten, PIO IN/OUT Module, Interface, Joysticks, Paddle, und...und...und...

Zubehör:
Kassetten, Disketten, Staubschutzhauben, Bücher, Bausätze, Resettaster, Alle Stecker, Bauteile, und...und...

Software:
Außergewöhnliche Spiele und Programme, z.B.: Flugsimulatoren, Dateiprogramme, Krankheitsdiagnose, Biorhythmus, Lottoberechnung und Spiele...Spiele...Spiele...

Prüfen Sie unser Angebot der Tiefstpreise.

„Schnell das neue Spitzeninfo anfordern, für 2 DM in Briefmarken. Garantiert 24 Std. Infoversand."

Händleranfragen erwünscht.

mükra DATEN-TECHNIK
Rotdornweg 15
1000 Berlin 45
☎ 030-817 38 57
341 45 73

SOFTWARE IM TEST

Burger Time

Graphik:

Sound:

Spielidee:

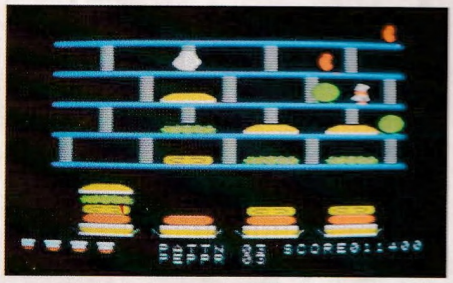
Testergebnis:

Eines der letzten Spiele, das von Texas Instruments als Modul herausgebracht wurde, ist Burger Time.

Der Held dieses Spieles ist Peter Pepper. Seine Aufgabe als Koch besteht darin, Hamburger zusammenzubauen. Dazu muß er in seiner Küche über verschiedene Ebenen laufen und die Zutaten in eine Schale hinunterschubsen. Ist Peter dann in einer Küche fertig, kann er gleich in der nächsten weitermachen.

Es soll in manchen Küchen Ratten und Mäuse geben. Die gibt es hier zwar nicht, trotzdem wird Peter ständig gestört. Drei unangenehme Zeitgenossen, Mr. Hot Dog, Mr. Pickle und Mr. Egg, versuchen, Peter Pepper zu Fall zu bringen. Doch mit einem Pfefferstreuer kann der jeweilige Gegener kurzzeitig außer Gefecht gesetzt werden. Noch besser ist es, ihn in einem Burger zu verarbeiten. Sollte Pepper dennoch erwischt werden, muß einer seiner fünf Partner die lebenswichtige Aufgabe des Burgerbauens übernehmen.

Leider ist dem Modul keine deutsche Anleitung beigelegt, aber dank einfachem Spielaufbau ist diese auch nicht unbedingt nötig.



Die Graphik ist ausgezeichnet, kommt aber nur mit einem Farbfernseher voll zur Geltung.

Das Spiel ist mit einer Melodie des Typs 'Ohrwurm' unterlegt, die nur unterbrochen wird, wenn eine Lage des Hamburgers fällt, einer der Störenfriede oder Peter Pepper lahmgelegt wird.

Der Schwierigkeitsgrad steigt mit jeder neuen Küche, so daß das Spiel auch nach längerem Einsatz nicht langweilig wird. Es könnte jedoch, ohne daß es hektisch würde, gerne etwas schneller sein.

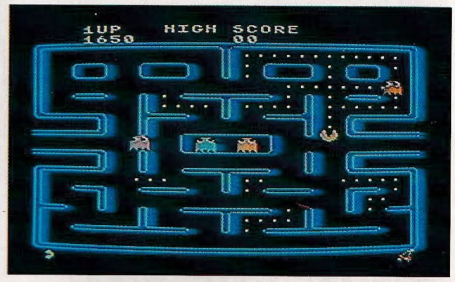
Pacman

Pacman ist wieder da. Die Handlung dieses wohl zu den berühmtesten der Gattung Computerspiele zählenden Spieles erklären zu wollen, hieße sicherlich, Eulen nach Athen tragen. Deshalb für Nichtspieler nur eine kurze Erklärung des Spielverlaufs: Der nimmersatte Pacman frißt sich durch ein Labyrinth, das mit Leckerbissen übersät ist. Verfolgt von vier Geistern muß er versuchen, alles Genießbare zu futtern, bevor er in die nächste Stufe vordringen kann. Mit Hilfe von vier Kraftpillen kann er jedoch für kurze Zeit aus Jägern Gejagte machen und die vier Gespenster hetzen. Leider läßt die Wirkung der Kraftpillen sehr schnell nach, und Pacman muß wieder fliehen.

Mit Pacman setzt Atari seine Reihe von Veröffentlichung bereits bekannter Spiele fort, die bisher nur den Atari-Besitzern zugänglich waren. Die Spielidee ist mittlerweile kaum noch neu zu nennen, da inzwischen viele mehr oder weniger gelungene Kopien des Originalpacmans auf dem Markt sind. Und sogar Texas Instruments, die für ihre Schnelligkeit in Bezug auf Softwareveröffentlichung keinesfalls bekannt sind, waren mit ihrem Munchman Jahre vorher auf dem Markt.

Der Sound ist wie im Original mittelmäßig, und auch die Graphik läßt den Glanz vergangener Tage nur hier und da in Form einer Erdbeere, versehen mit saftigen Zusatzpunkten, aufblitzen.

Dem echten Pacmanfan kann man dieses Modul, sofern er noch keinen Munchman hat, unbedingt empfehlen, denn es unterscheidet sich nicht merklich vom Original.



Ob man sich Pacman kaufen sollte, kann einem bei einem Preis von rund hundert Mark ganz allein der Blick in die eigene Brieftasche beantworten.

Graphik:

Sound:

Spielidee:

Testergebnis:

Bigfoot

Dieses Spiel führt uns in die schneebedeckten Gipfel der Anden. Als Bergsteiger versucht man, Goldvorräte und Proviantrationen von verschiedenen Felsvorsprüngen einzusammeln.

Weil diese aber weit auseinanderliegen, muß man sich eines Wurfankers und einer Strickleiter bedienen, um nach oben zu kommen. Hinunter kann man von einem Vorsprung zum anderen springen, aber auch seine bereits ausgehängten Strickleitern benutzen.

Das allein wäre natürlich sehr simpel und würde den Spieler zum Einschlafen bringen. Also dachten sich die Macher dieses Programms jemanden aus, der den Klettermaxen in Schwierigkeiten bringen soll: Bigfoot.

Dieses Yeti-Monster steht auf dem Gipfel des Berges und versucht, mit gigantischen Schneebällen den Bergsteiger ins Tal zu befördern. Gelingt ihm ein Treffer, so hat sich einer der drei zur Verfügung stehenden Kletterer abgeseilt — für immer!

Hat man alle Gold- und Proviantvorkommen gehamstert, so muß man er-

neut zum Gipfel hinauf und auf die Plattform von Bigfoot klettern. Dort angelangt, läßt man vom oberen Bildschirmrand einen Käfig hinunter auf Bigfoot und zieht ihn dann samt Bigfoot wieder hinauf. Schon hat man die nächste Stufe glücklich erreicht.

Die Spielidee ist mäßig. Nach einer gewissen Zeit hat man schon eine Routine. Lediglich die schneller werdenden Schneebälle verlangen schnelles Handeln. Für Baller-Spiel-Freaks ist dieses Spiel denkbar ungeeignet, benutzt doch der Bergsteiger nicht einmal ein Taschenmesser, um sich zu verteidigen, sondern ist voll und ganz in die Defensive gedrängt.

Die Graphik bleibt immer dieselbe, lediglich die Felsvorsprünge, auf denen sich die Güter befinden, ändern ihre Anordnung von Stufe zu Stufe und werden schwieriger zu erreichen.

Der Sound ist gut, besonders wenn der Kletterer seinen Wurfanker wirft, außerdem wird er noch durch den Sprachsynthesizer unterstützt.

Aufgrund des relativ langsamen Tempos kann das Spiel einigermaßen gut mit den Tasten gespielt werden.

Graphik:

Sound:

Spielidee:

Testergebnis:

Moon Patrol

Das aus den Spielhallen bekannte Spiel ist jetzt auch als Modul für den TI 99/4A zu bekommen.

Dem Spieler ist die Zone 9 zugeteilt worden, in der patrouilliert werden muß. Zu diesem Zweck ist ein modernes Fahrzeug bereitgestellt worden, das nicht nur fahren, sondern auch springen und schießen kann, denn Zone 9 ist so ziemlich die schlimmste: Der Boden ist mit Krater-Löchern übersät, Felsen stehen unmotiviert in der Landschaft rum, feindliche Aliens attackieren. Und dann hat auch noch irgendjemand den Weg zum Teil vermint!

Zwei verschiedene, in jeweils 26 Teile zerlegte Routen müssen kontrolliert werden. In jedem der 52 Teile lauern andere Gefahren, die bewältigt werden müssen. Es versteht sich, daß die Schwierigkeiten ständig zunehmen. Die aus den Spielhallen bekannte Graphik ist von Atari voll auf den TI 99/4A adaptiert worden. Die Anpassung geht soweit, daß auch die Ohrwurm-Dauermelodie den Spieler erfreut. Die Spielidee entstammt wohl dem alten "Defender"-Gedanken: "Töte oder du wirst getötet!"

Graphik:

Sound:

Spielidee:

Testergebnis:

Fortsetzung

Konnten wir euch beim letzten Mal das "99 Special I" aus dem TI-Learning Center ans Herz legen, haben wir es jetzt mit dem "99 Special II" zu tun.

Man kann das Buch als Fortsetzung des ersten Bandes betrachten: In gleichbleibender Qualität finden wir hier eine Fülle von Informationen und Listings vor, die aus dem TI 99/4A in den Sprachen TI BASIC, Extended BASIC, TMS9900 Assembler sowie UCSD-PASCAL das rausholen, was dieses hervorragende Gerät zu leisten vermag.

Zum Durcharbeiten der Beispiele sollte die vorhandene Konfiguration aus mindestens dem Grundgerät mit Erweiterungsbox, Disk-Controller, einem Laufwerk sowie den Moduln XBASIC und/oder Minimemory bestehen.

Mehr noch als beim ersten Band scheint die Hauptabsicht der Autoren darin zu liegen, weniger reine Technik als hauptsächlich Denkansätze dafür zu vermitteln, wie man logisch an Probleme und Aufgabenstellungen herangeht. Das zeigt sich schon im ersten Kapitel, das sich eingehend und anhand mehrerer Beispiele mit der Methode des Backtracking befaßt. Darauf folgt ein Abschnitt mit Betrachtungen und Programmen zur optimalen Datenorganisation.

Nach allerlei Nützlichem folgen die sogenannten Meisterspiele, von Kartenspielen bis zu "Cerberus".

Schade ist, daß die Autoren, die ja auch unter anderem ausgezeichnete Graphik-Software erstellen, sich nicht die Mühe gegeben haben, alle Graphiken und Pläne des Buches ansehnlich zu gestalten.

Wir halten "99 Special II" für mindestens genauso hervorragend wie Band I. Haben sollte man beide.

ThoK

99 Special II
Alma und Johann Peschetz
TI Learning Center
295/73831 ISBN 3-88078-045-5
476 Seiten
54,— DM

COMPUTER PRAXIS BASIC - PROGRAMMSCHEMA

PROGRAMM									BLATT	VON									
PROGRAMMIERER									DATUM										
4	8	12	16	20	24	28	32	36	40	44	48	52	56	60	64	68	72	76	80

The Bock Goes On!

Großes Echo erhielten wir auf unseren Blockgraphik-Artikel im Juli-Heft. Über die Zuschrift von Dirk Stockmann freuten wir uns besonders, so daß wir sie mit Abdruck und Honorar belohnen!

Liebe COMPUTER Praxis Redaktion!

Ich bin Schüler, 14 Jahre alt, und seit April Leser ihrer Zeitschrift.

Das Blockgraphikprogramm in ihrer letzten Ausgabe hat mir so gut gefallen, daß ich gleich nach einer praktischen Anwendungsmöglichkeit gesucht habe. Statt die Werte durch Zufallszahlen zu erzeugen, habe ich versucht, ein kurzes Programm für den TI zu schreiben, mit dem man verschiedene Statistiken eingeben kann. Leider ist es etwas länger geworden als geplant. Aber ich glaube, daß es trotzdem noch verwendbar ist.

Nun, über solche Leseraktivitäten freuen wir uns natürlich ungemein. Das Listing hat unsere kurze Routine in den Zeilen 320 bis 430 eingebaut. Die Änderungen in den Zeilen 320, 330, 360, 370 und 410 dienen zum einen der Benutzerfreundlichkeit (320/330), zum anderen dem Einlesen der gewünschten Daten (360—370 und 410). Die Definition der Zeichen und Farben wurde natürlich herausgenommen.

Zum TI-Listing:

Zeile

110—150: Initialisierung. WARNING wird ausgeschaltet, um unerwünschte Meldungen zu unterdrücken. Das Array W wird dimensioniert, die Zeichen definiert und die Farben gesetzt.

170—200: Menü. Vier verschiedene Routinen können aus diesem Menü angewählt werden: 1. Die Eingabe oder Änderung, 2. die Darstellung, 3. das Abspeichern und 4. das Einlesen einer Statistik. Die "5" unterbricht das Programm mit BREAK, es kann also mit CON fortgesetzt werden. Als Massenspeicher dient ein Cassetten-Recorder.

220—300: Eingabe/Änderung. Über die Sub-Routine ab Zeile 570 wird abgefragt, welche Statistik eingegeben, bzw. geändert werden soll. Das Array S$ nimmt den Namen auf, das Array W die Werte. Per Tastendruck wird dann die Statistik in einem Block-Diagramm ausgegeben.

320—430: Block-Graphik. Hier finden wir unseren Vorschlag aus Heft 7/84 wieder. Die Änderungen ermöglichen die Wiedergabe der eingegebenen Werte. Außerdem ist die Ablesbarkeit verbessert worden.

450—500: Abspeichern. In der ersten FOR-NEXT-Schleife wird die Anzahl der Statistiken ermittelt. Dann wird eine Cassetten-Datei mit einer Datensatz-Länge von 192 Zeichen eröffnet. Die Anzahl der Statistiken wird im ersten Satz ausgeladen, jeder folgende Satz beinhaltet die Daten einer ganzen Statistik! Dadurch wird die Schreib-Routine, und später auch die Lese-Routine, erfreulich kurz gehalten. Die Namen der Statistiken werden, zur Kontrolle, auf dem Bildschirm ausgegeben.

520—550: Einladen. Eine Cassetten-Datei zum Einladen wird entsprechend der Datensatz-Struktur der Ausladen-Routine eröffnet. Die Namen der eingelesenen Statistiken werden in S$, die Werte in W abgelegt. Die Namen werden zusätzlich auf dem Bildschirm fein säuberlich ausgegeben.

570—620: Sub-Routine "Inhaltsverzeichnis" und "Eingabetabelle". Der erste Teil der Sub-Routine gibt einen Namen der eingegebenen bzw. eingelesenen Statistiken wieder, der zweite Teil setzt die Optionen einer neuen Statistik, wie z.B. den Namen, die Skalen-Einteilung, und gibt die Monatswerte wieder. Die Sub-Routine wird aus den Routinen "Eingeben" und "Zeichnen" aufgerufen.

630: Unterbricht das Programm mit BREAK und setzt das Programm mit Zeile 110 fort, wenn das Kommando CON gegeben wird.

Die Variablen:

Arrays:	W(13,10)14*11 Plätze für die Werte. S$( ) undimensioniertes aber durch S in Zeile 580 auf 11 Plätze begrenztes Array für die Statistik-Namen.
Konstante:	A$,B$,C$,D$ Hex-Strings für die Zeichen-Definierung.
Hilfs-Var.:	K und ST für Tastatur-Abfragen, Z für die Höhe der Blöcke, ANZ in den Cassetten-Routinen für die Anzahl der Statistiken.
Schleifen:	I,M,S in den verschiedenen FOR-NEXT-Schleifen.

Achtung! Wer sich für eine Standard-BASIC-Version dieses Listings interessiert, schreibe uns bitte mit Rückumschlag. Wir schicken sie ihm zu.

H.U.

```
100 ! ZEICHENSATZ
110 ON BREAK NEXT :: ON WARNING NEXT ::
     CALL SCREEN(2):: CALL CLEAR :: DIM W(13,10)
120 CALL COLOR(10,5,2,11,10,2,12,1,10,13,4,1,14,10,
    4)
130 FOR I=0 TO 8 :: CALL COLOR(I,16,2):: NEXT I
140 A$="{16*(F)}" :: B$="0103070F1F3F7FFF" :: C$="
    {14*(F)}1F" :: D$="011A244A5224588"
150 CALL CHAR(91,D$,104,C$,112,A$,120,B$,128,B$,136,
    B$)
160 ! HAUPTMENUE
170 DISPLAY AT(10,1)ERASE ALL:
    "1.STATISTIK EINGEBEN/AENDERN": :
    "2.STATISTIK DARSTELLEN"
180 DISPLAY AT(14,1):"3.STATISTIK ABSPEICHERN": :
    "4.STATISTIK EINLESEN": :"5.ENDE"
190 CALL KEY(0,K,ST):: IF ST=0 OR K<49 OR K>53
    THEN 190
200 ON K-48 GOTO 220,320,450,520,630
210 ! WERTE EINGEBEN
```

TI BLOCK-DIAGRAMM XBASIC

```
220 GOSUB 570
230 ACCEPT AT(3,1)VALIDATE(UALPHA,NUMERIC)SIZE(-23)
    BEEP:S$(S)
240 ACCEPT AT(6,22)VALIDATE("12345")SIZE(-1)BEEP:
    W(0,S)
250 DISPLAY AT(9,8):"(MAXIMUM";W(0,S)*22;")"
260 W(13,S)=0 :: FOR M=1 TO 12
270 ACCEPT AT(M+10,15)VALIDATE(DIGIT)BEEP SIZE(-3):
    W(M,S):: IF W(M,S)>W(0,S)*22 THEN 270
280 W(13,S)=W(13,S)+W(M,S):: NEXT M ::
     W(13,S)=INT(W(13,S)/12)
290 DISPLAY AT(23,14):W(13,S)
300 CALL KEY(0,K,ST):: IF ST=0 THEN 300 ELSE 340
310 ! STATISTIK ZEICHEN
320 GOSUB 570
330 CALL KEY(0,K,ST):: IF ST=0 THEN 330
340 CALL CLEAR :: FOR I=22 TO 1 STEP -1 ::
     DISPLAY AT(24-I,1):I*W(0,S):: NEXT I
350 DISPLAY AT(24,4):"J F M A M J J A S O N D ["
360 FOR M=1 TO 13
370 Z=INT(W(M,S)/W(0,S))+1 :: P=(M*2)+4
380 IF Z<2 OR Z>23 THEN 410
390 CALL VCHAR(24-Z,P,104,Z)::
     CALL VCHAR(24-Z,P+1,112,Z)::
     CALL HCHAR(23,P+1,120)
400 CALL HCHAR(24-Z,P+1,136)::
     CALL HCHAR(24-Z,P,128)
410 NEXT M
420 DISPLAY AT(1,2):S$(S)
430 CALL KEY(0,K,ST):: IF ST=0 THEN 430 ELSE 170
440 ! ABSPEICHERN
450 FOR S=1 TO 10 :: IF S$(S)="" THEN 460 :: ANZ=S
460 NEXT S
470 CALL CLEAR :: OPEN #1:
    "CS1",INTERNAL,OUTPUT,FIXED 192
480 PRINT #1:ANZ :: PRINT :ANZ
490 FOR S=1 TO ANZ :: PRINT #1:
    S$(S),W(0,S),W(1,S),W(2,S),W(3,S),W(4,S),W(5,S),
    W(6,S),W(7,S),W(8,S),W(9,S),W(10,S),W(11,S),
    W(12,S),W(13,S)
500 PRINT :S$(S):: NEXT S :: CLOSE #1 :: GOTO 130
510 ! EINLESEN
520 CALL CLEAR :: OPEN #1:
    "CS1",INTERNAL,INPUT ,FIXED 192
530 INPUT #1:ANZ :: PRINT :ANZ
540 FOR S=1 TO ANZ :: INPUT #1:
    S$(S),W(0,S),W(1,S),W(2,S),W(3,S),W(4,S),W(5,S),
    W(6,S),W(7,S),W(8,S),W(9,S),W(10,S),W(11,S),
    W(12,S),W(13,S)
550 PRINT :S$(S):: NEXT S :: CLOSE #1 :: GOTO 130
560 ! INHALTSVERZEICHNIS
570 CALL CLEAR :: FOR S=1 TO 10 ::
     DISPLAY AT(S*2,1):S;".";S$(S):: NEXT S
580 DISPLAY AT(22,2):"PROGRAMMNR." ::
     ACCEPT AT(22,13)VALIDATE(DIGIT)BEEP:
    S :: IF S>10 THEN 580
590 ! EINGABETABELLE
600 DISPLAY AT(1,1)ERASE ALL:"NAME DER STATISTIK:":
     :S$(S): : :"SKALENEINHEIT (1-5):";W(0,S): : :
    "WERTE: "
610 DISPLAY AT(11,1):"JANUAR":"FEBUAR":"MAERZ":
    "APRIL":"MAI":"JUNI":"JULI":"AUGUST":
    "SEPTEMBER":"OKTOBER":"NOVEMBER":"DEZEMBER":
    "DURCHSCHNITT"
620 FOR M=1 TO 13 :: DISPLAY AT(M+10,14):
    W(M,S):: NEXT M :: RETURN
630 BREAK
640 GOTO 110
```

Das Atari-Listing BASIC

Für die Atari-Freunde haben wir die TI-Idee adaptiert. Die Namen der Statistiken werden in N$ gespeichert, je Name 10 Zeichen. Zeile 110 füllt N$ mit 10 mal "frei", damit ersichtlich ist, welche Statistiken nicht belegt sind. Die Zeile 120 ist nötig, da nach einem CLR Befehl dimensionierte Felder zwar nicht mehr ansprechbar, aber die letzten Werte noch vorhanden sind und bei neuer Dimensionierung nicht gelöscht werden. Ausgabe und Abfrage des Hauptmenüs folgen von 140 bis 200.

230—300: Neue Werte eingeben bzw. ändern. Im Subprogramm 660 wird eine Statistik ausgewählt, welche die Routine ab 690 auf dem Bildschirm ausdruckt. Die Eingaben werden mit der bei 750 beginnenden Routine unterstützt. Sie löscht die ausgedruckten Werte, wenn neue eingegeben werden. Sollen die alten Zahlen bleiben, braucht ihr nur RETURN zu drücken. Nachdem die Eingabe beendet ist, berechnet Zeile 290 den neuen Durchschnitt, PRINTet ihn als 13. Zahl und fragt, ob die Eingabe ok ist. Bei einem Fehler kann die Eingabe wiederholt werden, dazu 'N' tippen.

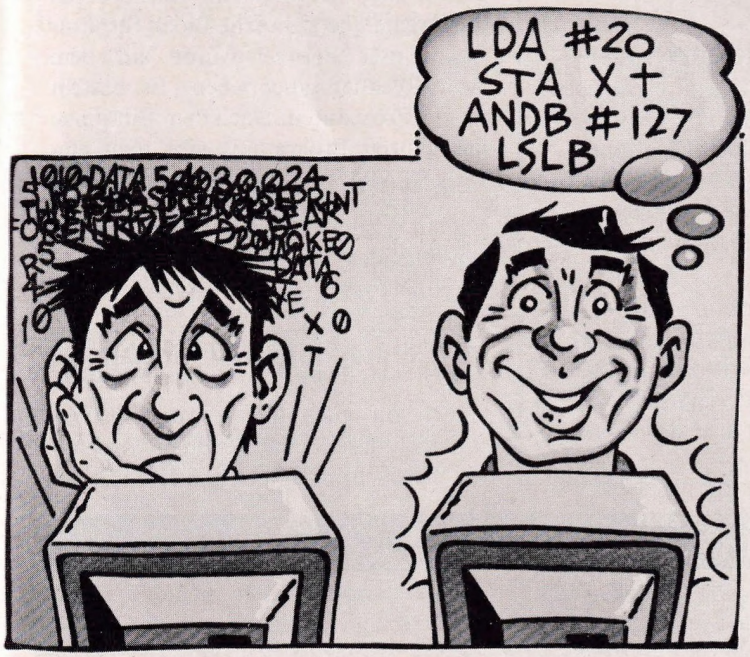
310—480: Die anzuzeigende Statistik wird wieder in den Subprogrammen 660 und 690 ausgewählt und ausgedruckt. Um nun das Blockdiagramm von den Zeilen 350 bis 470 zeichnen zu lassen, muß eine beliebige Taste gedrückt werden. Mit der ESC-Taste kommt ihr in das Hauptmenü zurück, jede andere Taste bewirkt einen Sprung in Zeile 330 und bringt wieder die Statistik auf den Schirm. Falls ihr versehentlich BREAK drückt, könnt ihr das Programm mit GOTO 150 ‹RETURN› neu starten. Ein Neustart mit RUN löscht die eingegebenen Werte!

490—630: Beim Abspeichern oder Laden von euren Statistiken werdet ihr nach GERÄT:DATEINAME gefragt. Als Gerät könnt ihr zwischen "C:" für Cassette oder "D:" für Diskette wählen. Der Dateiname ist nur bei Diskette anzugeben; dieser Name steht dann in der Directory.

660—780: Die Funktionen dieser Subprogramme wurde bereits erklärt, seht euch Zeile 710 mal genauer an ...

Falls beim Abspeichern oder Laden ein Fehler auftritt, wird ein Programm-Stop durch die Zeilen 790 bis 820 verhindert. Die Fehlernummer wird angezeigt, und ihr könnt die Bedeutung in eurem Handbuch nachlesen.

GB




```
100 DIM N$(100),W$(100),W(10,13)
110 N$="frei{6*(SPACE)}":N$(100)=" ":N$(11)=N$
120 FOR S=0 TO 9:FOR I=0 TO 13:W(S,I)=0:NEXT I:
    NEXT S
130 OPEN #5,4,0,"K"
140 REM HAUPTMENUE
150 POP
160 GRAPHICS 0:POKE 710,192:POKE 84,5
170 ? "{TAB}1...Statistik eingeben":
    ? "{TAB}2...Statistik darstellen"
180 ? "{TAB}3...Statistik abspeichern":
    ? "{TAB}4...Statistik einlesen":
    ? "{TAB}5...Ende":? :? "{TAB}?";
190 GET #5,KEY:ON KEY<49 OR KEY>53 GOTO 190
200 ON KEY-48 GOTO 240,320,490,570
210 REM ENDE
220 GRAPHICS 0:END
230 REM WERTE EINGEBEN
240 GOSUB 660:GOSUB 690
250 POSITION 12,3:GOSUB 750:INPUT W$
260 W$(11)="":N$(N,N1)=W$
270 W(S,13)=0:FOR I=1 TO 12
280 POSITION 18,6+I:GOSUB 750:TRAP 280:INPUT A:
    W(S,I)=A:W(S,13)=W(S,13)+A:NEXT I
290 W(S,13)=W(S,13)/12:POSITION 19,20:? W$:
    POSITION 19,20:? W(S,13):? "ok? (J/N)"
300 GET #5,KEY:ON KEY=74 OR KEY=155 GOTO 150:
    GOTO 250
310 REM WERTE ANZEIGEN
320 GOSUB 660:IF F THEN ? :
    ? "Programm nicht belegt!":GET #5,KEY:
    GOTO 150
330 GOSUB 690:? :? "{INV( ESC )} -> Menue ";:
    GET #5,KEY:IF KEY=27 THEN 150
340 REM BLOCKDIAGRAMM
350 GRAPHICS 7:POKE 752,1:SETCOLOR 2,7,13:
    SETCOLOR 1,3,6:SETCOLOR 0,13,6
360 ? "JA FE MA AP MA JU JL AU SE OK NO DE"
370 COLOR 3:FOR I=0 TO 79 STEP 4:PLOT 0,I:
    DRAWTO 2,I:IF INT(I/8)=I/8 THEN DRAWTO 5,I
380 NEXT I:MAX=0:FOR I=1 TO 12:
    IF W(S,I)>MAX THEN MAX=W(S,I)
390 NEXT I:W=MAX/77
400 FOR I=1 TO 12:Z=W(S,I):IF Z=0 THEN 470
410 FOR Q=-1 TO 3:COLOR 1:
    IF Z<W(S,13) THEN COLOR 2
420 PLOT I*12-Q,79-Z/W
430 DRAWTO I*12-Q,79:NEXT Q
440 COLOR 3:PLOT I*12+3,78
450 DRAWTO I*12+3,INT(77-Z/W+0.5)
460 DRAWTO I*12-2,INT(77-Z/W+0.5)
470 NEXT I:? "{TAB}100%=";MAX;"
      Durchsch.=";INT(W(S,13)*100)/100
480 ? " {INV( ESC )} -> Menue":GET #5,KEY:
    ON KEY=27 GOTO 150:GRAPHICS 0:POKE 710,192:
    GOTO 330
490 REM SPEICHERN
500 POKE 710,16:? "{CLEAR}{TAB}
    Statistik abspeichern"
510 POSITION 5,5:? "Geraet:Dateiname"
520 POSITION 9,6:INPUT W$:TRAP 790:
    OPEN #1,8,0,W$
530 FOR I=0 TO 9:IF N$(I*10+1,I*10+4)="frei"
    THEN 560
540 PUT #1,I:? #1;N$(I*10+1,(I+1)*10)
550 FOR Q=1 TO 13:? #1;W(I,Q):NEXT Q
560 NEXT I:PUT #1,100:CLOSE #1:GOTO 150
570 REM LADEN
580 POKE 710,16:? "{CLEAR}{TAB}Statistik laden"
590 POSITION 5,5:? "Geraet:Dateiname"
600 POSITION 9,6:INPUT W$:TRAP 790:
    OPEN #1,4,0,W$:? :? "   Lade:"
610 GET #1,S:IF S=100 THEN CLOSE #1:GOTO 150
620 INPUT #1,W$:N$(S*10+1,(S+1)*10)=W$
630 ? S,W$:FOR I=1 TO 13:INPUT #1,A:W(S,I)=A:
    NEXT I:GOTO 610
640 REM ENDE HAUPTPROGRAMM
650 REM SUB-PROGRAMME
660 ? "{CLEAR}":FOR I=0 TO 9:POSITION 8,5+I:
    ? I;"...";N$(I*10+1,(I+1)*10):NEXT I:
    POSITION 2,18:? "{DEL.LINE}{TAB}
    Programmnummer ";
670 TRAP 150:INPUT S:S=INT(S):TRAP 32767:
    ON S<0 OR S>9 GOTO 670:N=S*10+1:N1=N+9:
    F=(N$(N,N+3)="frei")
680 RETURN
690 ? "{CLEAR}":POSITION 2,3:? "Statistik ":?
700 IF  NOT F THEN POSITION 13,3:? N$(N,N1)
710 RESTORE 730:FOR I=1 TO 13:W$="{15*(.)}":
    READ W$:W$(15)=""
720 POSITION 5,6+I+(I=13):? W$;W(S,I):NEXT I:
    RETURN
730 DATA Januar,Februar,Maerz,April,Mai,Juni,
    Juli,August,September
740 DATA Oktober,November,Dezember,Durchschnitt
750 W$="{17*(SPACE)}":POKE 764,255:? "?";:
    X=PEEK(85)-1:Y=PEEK(84)
760 IF PEEK(764)=255 THEN 760
770 IF PEEK(764)<>12 THEN ? W$
780 POSITION X,Y:RETURN
790 ? :? "{BUZZER}Fehler- ";PEEK(195)
    ;" beim ";:IF KEY=51 THEN ? "Speichern";
800 IF KEY=52 THEN ? "Laden";
810 ? " von ";W$:CLOSE #1
820 ? :? "Weiter mit {INV( RETURN )}":
    GET #5,KEY:GOTO 150
```

Prozedur	1: GET. NUM	Menü-Eingabe
	2: MENUE	Hauptmenü
	3: DRAW.IT	Unter-Routine von PLOT.DATA, zeichnet die einzelnen Blöcke.
	4: ELEMENT	Holt aus einer gegebenen Liste den verlangten Wert.
	5: STAT.EIN	Unter-Routine von EINGEBEN. Hier werden die Werte der Statistik eingegeben.
	6: PLOT.DATA	Unter-Routine von DARSTELLEN. Stellt die Statistik als Block-Diagramm dar.
	7: WRT.SCALE	Zeichnet die Skala auf den rechten Rand des Bildschirms.
	8: DARSTELLEN	Ausgabe einer Statistik
	9: EINGEBEN	Eingabe einer Statistik
	10: START	Initialisierung des Programms
	11: D.SCHNITT	Errechnet den Durchschnitt der Statistik
	12: PLOT.DSCHNITT	Unter-Routine von DARSTELLEN, zeichnet den Durchschnitt in das Diagramm.

Das Atari-Listing LOGO

Für Freunde der Sprache LOGO haben wir dieselbe Aufgabe einmal anders gelöst. Das LOGO-Listing zeichnet sich durch eine übersichtliche Block-Struktur aus. Unser Beispiel wurde auf dem Atari-Rechner geschrieben. Es besteht aus 12 Prozeduren. Nach dem Eintippen des ganzen Programms gibt man einfach START ein, und schon läuft es.

Zum Listing geben wir euch im nebenstehenden Kasten eine Übersicht über die Prozeduren.

Commodore und Sinclair

Auch für euch haben wir selbstverständlich die Blockanwendung vorbereitet. Sie erscheint in unserem Heft 11/84.

BLOCK-DIAGRAMM LOGO

```
TO GET.NUM :MIN :MAX
MAKE "KEY ASCII RC MAKE "KEY :KEY - 48
IF ( AND :KEY > :MIN :KEY < :MAX ) [OP :KEY]
 [GET.NUM :MIN :MAX]
END

TO MENUE
CT PR [HAUPTMENUE] REPEAT 2 [PR[]]
PR [1...STATISTIK EINGEBEN]
PR [2...STATISTIK DARSTELLEN]
RUN ELEMENT GET.NUM 0 3 :RUNLISTE
END

TO DRAW.IT :N
PD FD :N * :SKALA BK :N * :SKALA
SETH 90 FD 1 SETH 0
FD :N * :SKALA BK :N * :SKALA
END

TO ELEMENT :N :LISTE
IF :N = 1 [OP FIRST :LISTE] [OP ELEMENT :N - 1 BF
 :LISTE]
END

TO STAT.EIN :N
IF :N > 12 [STOP]
SETCURSOR LIST 3 3 + :N
TYPE ELEMENT :N :MONAT
MAKE "NUM.IN RL MAKE "NUM.IN FIRST :NUM.IN
IF :NUM.IN > :MAX [MAKE "MAX :NUM.IN]
MAKE "NUM SE :NUM :NUM.IN STAT.EIN :N + 1
END

TO PLOT.DATA :N
IF :N > 12 [STOP]
PU SETPOS LIST ELEMENT :N :POSLIST -50 SETH 0
REPEAT 10 [DRAW.IT ELEMENT :N :NUM]
PLOT.DATA :N + 1
END

TO WRT.SCALE
REPEAT 41 [PD FD 4 BK 4 PU RT 90 FD 4 LT 90]
END
```

```
TO DARSTELLEN
CS HT CT PU
PR [JA FE MA AP MA JU JL AU SE OK NO DE]
SETPOS [-158 110] SETH 90 PD
WRT.SCALE PLOT.DATA 1
( PR [DURCHSCHNITT =] :DSCHNITT )
PLOT.DSCHNITT
PR [MIT RETURN GEHT'S WEITER]
MAKE "DUMMY RL STOP
END

TO EINGEBEN
TS CT MAKE "TOT 0 MAKE "MAX 0 MAKE "NUM []
STAT.EIN 1 D.SCHNITT :NUM
REPEAT 2 [PR []]
MAKE "SKALA 145 /:MAX STOP
END

TO START
RECYCLE
MAKE "RUNLISTE [[EINGEBEN] [DARSTELLEN]]
MAKE "MONAT [JANUAR.... FEBRUAR... MAERZ.....
 APRIL..... MAI{7*(.)} JUNI{6*(.)} JULI{6*(.)}
 AUGUST.... SEPTEMBER. OKTOBER... NOVEMBER..
 DEZEMBER..]
MAKE "POSLIST [-145 -124 -100 -78 -53 -27 -3 20
 45 69 92 117]
TS MENUE START
END

TO D.SCHNITT :LISTE
IF NOT EMPTYP :LISTE [MAKE "TOT :TOT +FIRST :LISTE
 D.SCHNITT BF :LISTE] [MAKE "DSCHNITT :TOT / 12
 STOP]
END

TO PLOT.DSCHNITT
PU SETPOS [-150 -50]
SETH 0 FD :DSCHNITT * :SKALA
SETH 90
REPEAT 50 [PD FD 3 PU FD 3]
END ■
```

Dr. Steven Molyneux

Spritzige Sprites

Nützlich für jedes Spiel: Wie man mit XBASIC bequem an die Sprites kommt, verrät euch Ronald Früchtnicht

Dieses Programm ermöglicht es euch, Sprites auf dem Bildschirm zu erzeugen. Die Besonderheit liegt darin, daß das auf der Schablone erzeugte Bild gleichzeitig als Sprite dargestellt wird. Dadurch entfällt das Problem, karierte Papierbögen vollzumalen, sich mit Hexadezimalcode herumzuschlagen und nachher festzustellen, daß das ganze im Original gar nicht so gut aussieht.

Hilfsvariablen (Laufvariablen):

X, Y, N1	— Arbeitsvariablen
ST	— Statusvariable für die CALL KEY Abfrage
K	— Variable für Feuertaste des Joysticks
K1	— Variable für die Tastaturabfrage
SP	— Joystick-Position auf der Spalte (1—256)
ZE	— Joystick-Position auf der Zeile (1—192)
Z1	— Aus ZE errechnete Zeile (8—23)
SP1	— Aus SP errechnete Spalte (8—23)
Q	— weist auf das entsprechende Tabellenelement von HEX und HEX$ (1-64)
CHDEF$	— beinhaltet den 64stelligen CALL CHAR Definitionswert.
A$, B$	— Hilfsstrings
I, J	— Schleifenvariablen

Zur Anwendung

Nach dem Programmstart wird das Laden der Tabellen durch eine knappe Darstellung der Programmfunktionen überbrückt. Der Befehl CALL MATRIX holt die Zeichenschablone mit 16 x 16 Feldern auf den Bildschirm. Nun kann 'losgemalt' werden: Durch Druck auf die Feuertaste wird ein Punkt gesetzt. Die Praxis hat gezeigt, daß man häufig symmetrische Sprites benötigt. Die Funktion "S" bildet die Zeichen der linken Seite des Sprites auf der rechten Seite symmetrisch ab.

Hat man mal einen Punkt falsch gesetzt, so löscht man ihn durch Drücken der Taste "l". Nachdem man seinen Sprite fertig hat, betätigt man die Leertaste. Das Sprite wird nun abgespeichtert. Wenn der Rechner wieder bereit ist, erscheint im Display: "Ende (J/N)":

▷ N: Die Zeichenschablone wird 'saubergemacht', und man kann weitere Sprites erzeugen.

▷ J: Es erscheint Bild 2, und sämtliche bisher erzeugten Sprites werden dargestellt. Es besteht nun die Auswahl aus 4 Funktionen (N-W-P-D):

▷ N, Neu: Alle bisher erzeugten Sprites werden gelöscht; das Programm verzweigt zu Bild 1.

▷ W, Weiter: Die bisher erzeugten Sprites bleiben erhalten; das Programm verzweigt zu Bild 1.

▷ P, Pattern: Die bisher erzeugten Sprites werden mit der Patternfunktion übereinander dargestellt. Es entsteht der Eindruck von Bewegung.

▷ D, Darstellung: Die erzeugten Sprites können durch Drücken der Leertaste mit der zugehörigen Char-Definition abgerufen werden.

Hat man Funktion 4 gewählt und alle Sprites durchgeblättert, so steht es einem frei, dies zu wiederholen (N =

Nochmal), weitere Sprites zu bauen (W = Weiter) oder seine Arbeit zu beenden (E = Ende). Die Ende-Funktion ist gegen versehentliche Benutzung abgesichert. Nach "CON" (Continue) nimmt der Rechner die Arbeit ohne Datenverlust wieder auf.

Alles in allem haben wir festgestellt, daß das Erstellen von Sprites reichlich Spaß macht, und außerddem sehen die Raketen, Monster und Männchen so aus, wie man es sich für seine Spielidee wünscht.

Benutzte Variablen:

Tabellen:

Name	anz. Felder	Beschreibung
HEX	64	dez.Werte (0 — 15)
HEX$	64	Hexawerte im String-Format
AVZE	256(16x16)	enthält die Werte 8,4,2,1 waagerecht angeordnet
MAT	256(16x16)	enthält der Werte 1 — 64 (jeder Wert 4x) horizontal zweizeilig.
SYM	16	enthält die entsprechenden Hexawerte für die Symmetriedarstellung
SPRITE$	20	dient der Speicherung der Spritedefinitionen

```
100 CALL CLEAR :: CALL SCREEN(2)
110 FOR I=2 TO 8 :: CALL COLOR(I,5,1):: NEXT I
120 ZEI=96 :: SPR=1
130 DISPLAY AT(1,1):"{6*(SPACE)}
    SPRITE  EDITOR{44*(SPACE)}FUNKTIONEN"
140 DISPLAY AT(6,1):"JOYST1 = STEUERUNG{38*(SPACE)}
    FEUERT = ZEICHNEN"
150 DISPLAY AT(10,1):"LEERT. = SPRITE SPEICHERN
    {31*(SPACE)}L-TASTE= ZEICHENLOESCHUNG"
160 DISPLAY AT(14,1):"L+ALPHA LOCK = GES.LOESCH
    {31*(SPACE)}S= SYMETRIE  RECHTE HAELFTE=
    {13*(SPACE)}LINKE  HAELFTE"
170 OPTION BASE 1
180 DIM HEX(64),HEX$(64),AVZE(16,16),MAT(16,16),
    SPRITE$(20)
190 FOR I=1 TO 16 :: FOR J=1 TO 16
200 X=X+1 :: IF X=5 THEN X=1 :: RESTORE
210 READ AVZE(I,J)
220 NEXT J :: NEXT I
230 DATA 8,4,2,1
240 ZAHL=1
250 FOR I=1 TO 16 :: FOR J=1 TO 8
260 N1=N1+1 :: IF N1=5 THEN ZAHL=ZAHL+1 :: N1=1
270 MAT(I,J)=ZAHL
280 NEXT J :: NEXT I
290 DISPLAY AT(5,1):"{9*(*)}  BILD 2  {9*(*)}
    {28*(SPACE)}N = NEU,BESTEHENDE SPRITES
    {6*(SPACE)}WERDEN GELOESCHT "
300 DISPLAY AT(10,1):"W = WEITER,SPR.BLEIBEN ERH.
    {29*(SPACE)}P = PATTERN,BEWEGUNG ERZ."
310 DISPLAY AT(14,1):"D = DARSTELLUNG MIT CHAR.
    {7*(SPACE)}DEFINITION{74*(SPACE)}"
320 N1=0 :: ZAHL=ZAHL+1
330 FOR I=1 TO 16 :: FOR J=9 TO 16
340 N1=N1+1 :: IF N1=5 THEN ZAHL=ZAHL+1 :: N1=1
350 MAT(I,J)=ZAHL
360 NEXT J :: NEXT I
370 DIM SYM$(16)
380 FOR I=1 TO 16
390 READ SYM$(I):: NEXT I
400 DATA 0,8,4,C,2,A,6,E,1,9,5,D,3,B,7,F
410 CHDEF$=RPT$("0",64)
420 IND=1
430 DISPLAY AT(24,4):"BELIEBIGE TASTE DRUECKEN" ::
     CALL KEY(0,K,ST):: IF ST=0
    THEN 430 ELSE CALL CLEAR
440 CALL MAGNIFY(3)
450 !{21*(*)}
460 CALL MATRIX !{9*(*)}
470 !{21*(*)}
480 !
490 CALL JOYST(1,X,Y):: ZE=ZE+-Y*2 :: SP=SP+X*2
500 IF SP<56 THEN SP=56
510 IF SP>176 THEN SP=176
520 IF ZE<56 THEN ZE=56
530 IF ZE>176 THEN ZE=176
540 CALL LOCATE(#1,ZE,SP)
550 CALL KEY(1,K,ST):: CALL KEY(0,K1,ST)
560 IF K=18 THEN 620
570 IF K1=32 THEN 770
580 IF K1=108 THEN 970
590 IF K1=76 THEN 1010
600 IF K1=115 THEN 1180
610 GOTO 490
620 Z1=(ZE+8)/8 :: SP1=(SP+8)/8 ::
     CALL GCHAR(Z1,SP1,XX):: IF XX=60 THEN 490
630 Q=MAT(Z1-7,SP1-7):: HEX(Q)=HEX(Q)
    +AVZE(Z1-7,SP1-7):: CALL HCHAR(Z1,SP1,60)
640 IF HEX(Q)<10 THEN 720
650 ON HEX(Q)-9 GOSUB 660,670,680,690,700,710 ::
     GOTO 720
660 HEX$(Q)="A" :: RETURN
670 HEX$(Q)="B" :: RETURN
680 HEX$(Q)="C" :: RETURN
690 HEX$(Q)="D" :: RETURN
700 HEX$(Q)="E" :: RETURN
710 HEX$(Q)="F" :: RETURN
720 A$=SEG$(CHDEF$,1,Q-1)::
     B$=SEG$(CHDEF$,Q+1,64-Q+1)
730 IF HEX(Q)>9 THEN CHDEF$=A$&HEX$(Q)&B$ ::
     GOTO 750
```


ACHTUNG!
Bei uns finden Sie
fast alles für Ihren
VC-20/64

Hardware:
Speichererweiterungen 8K-64K RAM, Moduladapter 2/3/5-fach, Recorderinterface, Schnell-Save Module, Toolkitmodule, 40/80 Zeichenkarten, Epromprogrammierer, Epromkarten, PIO IN/OUT Module, Interface, Joysticks, Paddle, und...und...und...

Zubehör:
Kassetten, Disketten, Staubschutzhauben, Bücher, Bausätze, Resettaster, Alle Stecker, Bauteile, und...und...

Software:
Außergewöhnliche Spiele und Programme, z.B.: Flugsimulatoren, Dateiprogramme, Krankheitsdiagnose, Biorhythmus, Lottoberechnung und Spiele...Spiele...Spiele...

Prüfen Sie unser Angebot der Tiefstpreise.

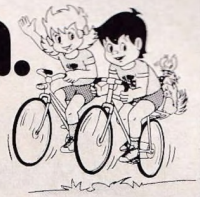
„Schnell das neue Spitzeninfo anfordern, für 2 DM in Briefmarken. Garantiert 24 Std. Infoversand."

Händleranfragen erwünscht.

mükra DATEN-TECHNIK
Rotdornweg 15
1000 Berlin 45
☎ 030-817 38 57
341 45 73

```
740 CHDEF$=A$&STR$(HEX(Q))&B$
750 CALL CHAR(92,CHDEF$)::
     CALL SPRITE(#2,92,8,15,115):: GOTO 490
760 !{51*(*)}
770 IF CHDEF$=RPT$("0",64)
    THEN 780 ELSE SPR=SPR+1 ::
     SPRITE$(SPR)=CHDEF$ :: CHDEF$=RPT$("0",64)
780 FOR I=1 TO 64 :: HEX(I)=0 :: HEX$(I)=" " ::
     NEXT I :: CALL MATRIX
790 DISPLAY AT(24,8):"ENDE ? (J/N)" ::
     CALL KEY(0,K,ST):: IF ST=1
    THEN DISPLAY AT(24,8):" " ELSE 790
800 IF K=106 THEN 820 !J
810 IF K=110 THEN 460 ELSE 790 !N
820 CALL CLEAR :: CALL DELSPRITE(ALL)
830 S=80 :: Z=0
840 FOR I=2 TO SPR :: Z=Z+40 :: SZ=SZ+1 :: IF SZ=5
    THEN S=S+40 :: Z=40 :: SZ=0
850 ZEI=ZEI+4 :: CALL CHAR(ZEI,SPRITE$(I))::
     CALL SPRITE(#I,ZEI,8,Z,S):: NEXT I :: ZEI=96
860 DISPLAY AT(2,5):"N*W*P*D" :: CALL KEY(0,K,ST)::
     IF ST=1 THEN DISPLAY AT(2,5):" " ELSE 860
870 IF K<>110 THEN 890 !N
880 CALL DELSPRITE(ALL):: FOR I=2 TO SPR ::
     SPRITE$(I)=" " :: NEXT I :: SPR=1 :: GOTO 460
890 IF K=119 THEN CALL DELSPRITE(ALL):: GOTO 460 ! W
900 IF K=100 THEN GOTO 1030
910 IF K<>112 THEN 860 !P
920 CALL DELSPRITE(ALL):: CALL
     SPRITE(#10,100,8,90,120)
930 FOR OFT=1 TO 10
940 FOR I=100 TO 92+SPR*4 STEP 4 ::
     CALL PATTERN(#10,I):: CALL JOYST(1,X,Y)::
     CALL MOTION(#10,-Y*2,X*2):: NEXT I
950 NEXT OFT :: CALL MOTION(#10,0,0)::
     CALL LOCATE(#10,90,120):: GOTO 860
960 !{22*(*)}{6*(SPACE)}loesch
970 Z1=(ZE+8)/8 :: SP1=(SP+8)/8
980 CALL GCHAR(Z1,SP1,XX):: IF XX<>60 THEN 490
990 Q=MAT(Z1-7,SP1-7):: HEX(Q)=HEX(Q)
    -AVZE(Z1-7,SP1-7):: CALL HCHAR(Z1,SP1,34)
1000 GOTO 640
1010 !GESAMTLOESCH
1020 FOR I=1 TO 64 :: HEX(I)=0 :: HEX$(I)=" " ::
     NEXT I :: CALL MATRIX :: CHDEF$=RPT$("0",64)::
     GOTO 490
1030 CALL DELSPRITE(ALL):: FOR I=2 TO SPR
1040 DISPLAY AT(12,7):"NR.";I-1
1050 CALL SPRITE(#I,92+4*I,16,87,125)
1060 DISPLAY AT(14,5):"DEFINITION :{43*(SPACE)}
    ";SPRITE$(I)
1070 DISPLAY AT(24,1):"    WEITER =  LEERTASTE"
1080 CALL KEY(0,K,ST):: IF ST=0 THEN 1080
1090 CALL DELSPRITE(#I):: NEXT I
1100 DISPLAY AT(24,1)ERASE ALL:
    "NOCHMAL=N WEITER=W ENDE=E"
1110 CALL KEY(0,K,ST):: IF ST=0 THEN 1110
1120 DISPLAY AT(24,1):""
1130 IF K=119 THEN CALL CLEAR :: GOTO 460
1140 IF K=110 THEN 1030
1150 IF K<>101 THEN 1100
1160 DISPLAY AT(12,1)ERASE ALL:
    "PROG.ENDE RESTART MIT 'CON'{37*(SPACE)}
    MOEGLICH" :: BREAK
1170 CALL CLEAR :: CALL SCREEN(2):: FOR I=2 TO 8 ::
     CALL COLOR(I,5,1):: NEXT I :: GOTO 440
1180 ! SYMETRIE
1190 FOR I=1 TO 31 STEP 2
1200 HEX$(I+33)=SYM$(HEX(I)+1)
1210 NEXT I
1220 FOR I=2 TO 32 STEP 2
1230 HEX$(I+31)=SYM$(HEX(I)+1)
1240 NEXT I
1250 CHDEF$=SEG$(CHDEF$,1,32)
1260 FOR I=33 TO 64
1270 CHDEF$=CHDEF$&HEX$(I)
1280 NEXT I
1290 GOTO 750
1300 SUB MATRIX
1310 CALL CHAR(33,"0101010101010101")
1320 CALL COLOR(1,5,1)
1330 CALL CHAR(34,"01010101010101FF")
1340 CALL CHAR(35,"{14*(0)}FF")
1350 CALL CHAR(60,"{16*(F)}")
1360 CALL CHAR(96,"181818FFFF181818{48*(0)}")
1370 ZE=104 :: SP=128
1380 CALL SPRITE(#1,96,11,1,1)
1390 FOR Z=8 TO 23
1400 CALL HCHAR(Z,8,34,16)
1410 CALL HCHAR(Z,7,33)
1420 CALL HCHAR(7,Z,35)
1430 NEXT Z
1440 CALL HCHAR(15,7,34)
1450 CALL HCHAR(24,15,33)
1460 CALL HCHAR(7,15,34)
1470 CALL HCHAR(15,24,35)
1480 SUBEND ■
```


Bring den Puls auf 130! Beim Radfahren.



© DSB/SB 1984

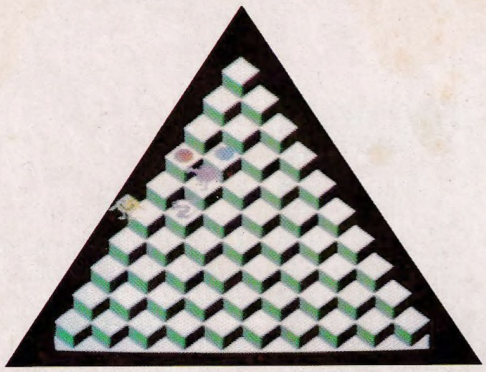
Suchen Sie die Idee für ein aktives Wochenende? Machen Sie mit der Familie eine Radtour ins Grüne! Welches Tempo angemessen ist, sagt Ihnen die Formel Trimming 130. Damit kann jeder sein Tempo bestimmen. Schon 10 Minuten täglich Trimming genügen, um Herz und Kreislauf zu trainieren, wenn das Herz dabei etwa 130 Pulsschläge in der Minute erreicht. Aber auch dort, wo Radfahren nur schwer möglich ist, kann man mit Trimming 130 fit werden. In vielen Sportarten, beispielsweise mit Laufen, Gymnastik, Schwimmen oder Kanufahren.

Wie mißt man den Puls? Ganz einfach! Pause einlegen, Puls fühlen, mit Hilfe einer Armbanduhr Schläge in 10 Sekunden zählen. Wenn Sie 21, 22 oder 23 Pulsschläge zählen, sind Sie im Richtmaß Trimming 130. Die Broschüre „Gesundheit durch Trimming 130" gibt es gegen DM –,80 Porto vom Deutschen Sportbund, Postfach, 6000 Frankfurt/Main 71.

trimm 130 Bewegung ist die beste Medizin

Trimm Dich durch Sport

dsb Deutscher Sportbund



TI-BERT

Ihr springt im Dreieck
X-BASIC

Das hier vorgestellte Spiel hat den berühmten QBert zum Vorbild bild. Ihr wißt schon: das kleine Rüsselmännchen, das von Würfel zu Würfel springt. Klaus Weidemann hat das Spiel für den TI adaptiert.

Wenn ihr das Spiel eingeben wollt, braucht ihr das Extended BASIC-Modul. Ein wenig Phantasie wäre auch angebracht, denn wir stellen TI-Bert in einer Spielstruktur vor, die zwar lauffähig ist, aber von euch weiter ausgebaut werden kann und soll. Aus diesem Grund wurde die Möglichkeit einbezogen, eigene Subs zu definieren. Wenn ihr also eigene Ideen habt, um das Spiel zu erweitern, nur ans Werk! Kaum nötig zu sagen, daß wir an euren Weiterentwicklungen sehr interessiert sind!

Das Ziel des Spiels ist, alle Würfelflächen von TI-Bert einmal anspringen zu lassen. Daß ihr dabei nicht in Ruhe gelassen werdet, versteht sich von allein. Wir empfehlen euch, das Spiel nicht mit den Original-Joysticks zu spielen. Nehmt bessere, sonst ärgert ihr euch grün und blau. Exakte Steuerungen sind mit den TI-Joysticks nämlich kaum möglich, was zur Folge hat, daß TI-Bert des öfteren übermütig von einer Würfelpyramide springt und eines seiner vier Leben verliert.

Achtung, wenn ihr das Spiel ausbaut, dann achtet auf einige Kleinigkeiten. Alle Sprites bestehen aus vier Charakters. Die Variablen LE, RU und Z müssen an die eventuell neu geschaffenen Subs übergeben werden, da das Spiel sonst nicht richtig funktioniert. Wenn das Spiel beendet werden soll, dann bitte über die Subroutine Sprungzähler, da sonst das Spielgeschehen unnötig verlangsamt wird. Die notwendige Abfrage sollte in Zeile 410 erfolgen.

Die Erfolgsmelodie

Kleiner Tip am Rande: Um die Erfolgsmelodie auf Eingabefehler zu überprüfen, ohne dazu das gesamte Spiel durchspielen zu müssen, kann man die Subroutine probeweise folgendermaßen aufrufen:

```
1 CALL ERF (U1,LE,RU,Z)
2 GOTO 1
```

Die beiden Zeilen können vor dem Hauptprogramm stehen und bei intakter Erfolgsroutine gelöscht werden.

Und nun wünschen wir euch viel Spaß und Kreativität beim Ausbau des TI-Bert.

Variablen:

Z	Sprungzähler
LE	Anzahl der vorhandenen Leben vom TI-Bert
RU	Rundenzähler

Sprites:

1	TI-Bert
2	TI-Berts Comicfluch
3,4,5,6	TI-Berts Gegner: zwei Coily, Sam der Arbeitsvernichter, die Springerschlange.

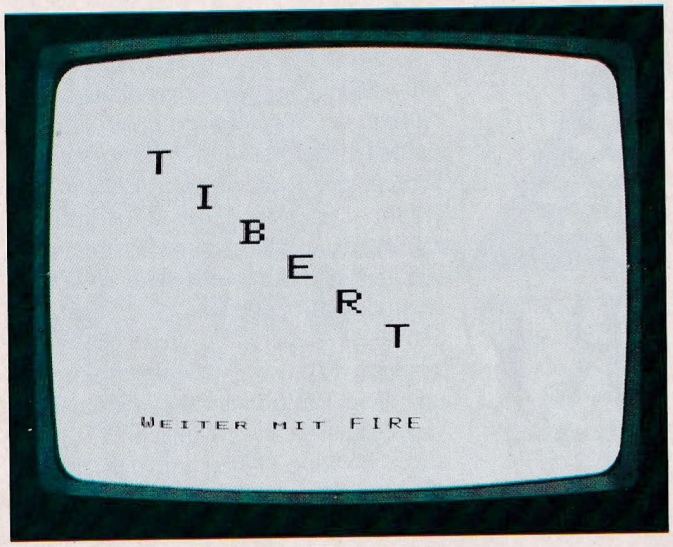
Struktur:

100—110 setzt Z, LE und RU auf Null.
130 setzt FE auf Null
140—230 definiert die für das Spiel notwendigen Charakter.
230—240 setzt die Farben.
250 ruft die Würfelgraphik auf
260—300 Das Herzstück des Programms. Hier werden der Joystick 1 abgerufen und die Bewegungsroutinen aufgerufen.
310 ruft die Feindroutine auf.
320—340 überprüft, ob die Feinde im gültigen Bereich sind, sonst werden sie gelöscht.
350 Rücksprung zur Joystickabfrage.
380—420 Der Zählmechanismus: Die Variable V1 zeigt an, von wo das Zählwerk angesprochen wird. Wenn V1 = 1 ist, erfolgt der Aufruf von der Erfolgs-, sonst von den Bewegungsroutinen. Dementsprechend erfolgt bei V1 = 1 die Verzweigung nach Subexit, sonst zur Würfelgraphik.
440—560 Die Würfelgraphik. Beim Aufbau der Würfelgraphik wird die Variable LE auf 4, Z auf Null gesetzt, außerdem der Rundenzähler RU um eins erhöht. Der Aufruf der Würfelgraphik erfolgt genau dann, wenn TI-Bert seine vier Leben verloren oder alle Würfelflächen einmal berührt hat.
580—680 steuert die Feindbewegung. TI-Berts Position wird abgefragt, und um diese Position herum werden seine vier Gegner gesetzt. Danach wird abgefragt, ob sich TI-Bert mit den anderen Sprites trifft. Wenn ja, wird die Absturzroutine aufgerufen.
700—960 Hier finden wir die Bewegungsroutinen für TI-Bert. Wie im Original kann sich TI-Bert nur diagonal bewegen. Dementsprechend sind die Joystickabfragen mit den Bewegungsroutinen aufeinander abgestimmt.

Pixellogik

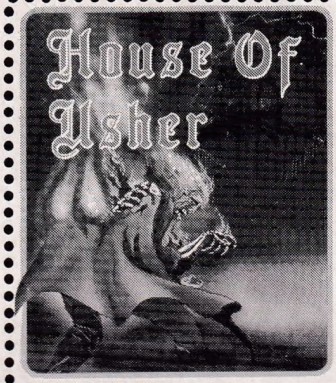
Sub LU steuert TI-Bert nach links unten, Sub LO nach links oben, Sub RO nach rechts oben und RU nach rechts unten. In diesen Routinen wird ständig TI-Berts Position abgefragt. Springt TI-Bert von seiner Pyramide herunter, erfolgt sofort der Aufruf der Absturzroutine. Springt TI-Bert auf ein gültiges Feld, wird überprüft, ob auf dem Feld eine Pixelmarkierung gesetzt ist. Wenn nein, dann wird sie gesetzt, das Zählwerk aufgerufen und Z um eins erhöht.
980—1180 Spielbeschreibung.
1200—1280 TI-Berts Absturzroutine. Mit jedem Aufruf dieser Routine wird die Variable LE um Eins verringert. Hat LE den Wert Null erreicht, ist die jeweilige Spielrunde beendet und die Würfelgraphik wird abgerufen. Alle gesetzten Pixel werden gelöscht. Die Absturzroutine steuert das 'sich Schütteln' von TI-Bert und setzt seinen Comicfluch an die richtige Stelle.
1300—1490 Hat TI-Bert alle Würfelflächen einmal berührt, wird vom Zählwerk die Erfolgsroutine abgerufen, die in diesen Zeilen untergebracht ist. Es werden alle Feindsprites gelöscht, nur TI-Bert bleibt in seiner letzten Position stehen und lauscht den schönen Klängen. Dann beginnt das Spiel von vorn.

K.W.




```
100 Z=0 :: CALL SPIEL
110 LE=4 :: RU=0
120 CALL CLEAR
130 FE=0
140 K$="{9*(0)}3070F0F0F0F0703{16*(0)}
    C0E0F0F0F0F0E0C0"
150 A$="0103070F1F3F7FFF" ::
     B$="80C0E0F0F8FCFEFF" :: D$="8183878F1F3F7FFF"
160 Q$="0143757F4F0707070703010700000100C0F0F87CFCFC
    FCFCF8F060404040C0"
170 Q1$="030F3F3E3F3F3F3F1F0F060202020300800C2AEFEF2F
    0E0E0E0C080E0000080"
180 Q2$="{7*(0)}1033F6075ED6E3F0303010101{6*(0)}
    F0FCE6BBA7EFBCE08080"
190 S$="000103070D1F1B3B6DCE070202020E000080C0E0B0F8
    D8DCB673E040404070"
200 SP$="{9*(0)}71C10161F1803030707030000182870E0000
    0C0E060C4C810E0C0"
210 CALL CHAR(40,Q$,44,Q1$,80,Q2$,84,K$,88,K$,60,S$,
    36,SP$)
220 CALL CHAR(113,D$)
230 CALL CHAR(104,A$,112,A$,120,A$,128,B$,136,B$,96,
    B$):: CALL COLOR(10,16,1,11,2,16,12,16,2,13,16,
    1,14,3,16,9,16,3)
240 CALL COLOR(3,16,2,4,16,2)
250 CALL GRAPH(LE,RU,Z,A)
260 CALL MAGNIFY(3):: CALL SPRITE(#1,40,14,85,97)
270 CALL JOYST(1,X,Y):: CALL MOTION(#1,-Y/16,X/16)::
     IF X=-4 AND Y=-4 THEN CALL LU(X1,Y1,Z,LE,RU)
280 IF X=-4 AND Y=4 THEN CALL LO(X1,Y1,Z,LE,RU)
290 IF X=4 AND Y=4 THEN CALL RO(X1,Y1,Z,LE,RU)
300 IF X=4 AND Y=-4 THEN CALL RU(X1,Y1,Z,LE,RU)
310 CALL FE(FE,F,LE,RU,Z)
320 CALL POSITION(#3,X,Y):: IF X>200 OR X<40
    THEN CALL DELSPRITE(#3)
330 CALL POSITION(#4,X,Y):: IF X>200
    THEN CALL DELSPRITE(#4)
340 CALL POSITION(#6,X,Y)::
     IF X>200 OR X<41 OR Y>190
    THEN CALL DELSPRITE(#6):: FE=0
350 GOTO 270
360 SUB Z(Z,A,LE,RU)
370 !
380 ! SPRUNG-ZAEHLER
390 U1=0 :: Z=Z+1 :: IF Z=55
    THEN CALL ERF(U1,LE,RU,Z)
400 IF U1=1 THEN 410 ELSE SUBEXIT
410 CALL CLEAR :: CALL SCREEN(2)::
     CALL GRAPH(LE,RU,Z,A)
420 SUBEND
430 !
440 ! WUERFELGRAPHIK
450 SUB GRAPH(LE,RU,Z,A)
460 Z=0 :: A=1 :: K=6 :: K1=24 :: LE=4 :: RU=RU+1 ::
     DISPLAY AT(3,2):"runde:";RU :: DISPLAY AT(2,2):
    "LEBEN:";LE
470 FOR P1=24 TO 1 STEP -2
480 FOR S=K TO K1 STEP 2
490 P=96 :: CALL HCHAR(P1,S,P):: P=120 ::
     CALL HCHAR(P1,S+1,P):: P=96 :: NEXT S
500 FOR S=K TO K1 STEP 2
510 P=136 :: CALL HCHAR(P1-1,S,P):: P=112 ::
     CALL HCHAR(P1-1,S+1,P):: P=136 :: NEXT S
520 K=K+1 :: K1=K1-1
530 NEXT P1
540 S=6 :: U=25
550 FOR P1=22 TO 3 STEP -2 :: P=104 ::
     CALL HCHAR(P1,S,P):: P=128 ::
     CALL HCHAR(P1,U,P):: P=104 :: S=S+1 :: U=U-1 ::
     NEXT P1
560 SUBEND
570 !
580 ! FEINDBEWEGUNG
590 SUB FE(FE,F,LE,RU,Z)
600 CALL POSITION(#1,X1,Y1)
610 IF FE<>0 THEN 660
620 CALL SPRITE(#3,84,7,X1-13,Y1-8,5.5,-3)
630 CALL SPRITE(#4,88,6,X1-14,Y1+9,8.1,4.4)
640 CALL SPRITE(#5,60,11,X1+16,Y1-32,0,10)
650 CALL SPRITE(#6,36,14,X1+16,Y1-8,5.5,3)
660 CALL COINC(ALL,X)::IF X THEN FE=1 ::
     CALL AB(LE,X1,Y1,U,Z,RU)
670 FE=FE+1 :: SUBEXIT
680 SUBEND
690 !
700 ! BEWEGUNG TIBERT
710 SUB LU(X1,Y1,Z,LE,RU)
720 CALL PATTERN(#1,40):: CALL POSITION(#1,X1,Y1)::
     CALL GCHAR((X1+16)/8,(Y1+16)/8,AC)::
     IF AC<>113 THEN CALL HCHAR((X1+16)/8,(Y1+16)/8,
     113)::CALL Z(Z,A,LE,RU)
730 IF X1>163 THEN CALL AB(LE,X1,Y1,U,Z,RU)ELSE 750
740 SUBEXIT
750 FOR I=1 TO 16 STEP 4 ::
     CALL LOCATE(#1,X1+I+3,Y1-8):: NEXT I ::
     CALL SOUND(30,-7,5,150,5)
760 SUBEND
770 SUB LO(X1,Y1,Z,LE,RU)
780 U=0 :: CALL PATTERN(#1,40)::
     CALL POSITION(#1,X1,Y1)::
     CALL GCHAR((X1+16)/8,(Y1+16)/8,AC)
790 IF AC<>113 THEN CALL HCHAR((X1+16)/8,(Y1+16)
    /8,113):: CALL Z(Z,A,LE,RU)
800 CALL GCHAR((X1-8)/8,Y1/8,AC):: IF AC=32
    THEN CALL AB(LE,X1,Y1,U,Z,RU)
810 IF U=1 THEN 830
820 FOR I=1 TO 8 STEP 2 ::
     CALL LOCATE(#1,X1-16,Y1-I-1):: NEXT I ::
     CALL SOUND(30,-7,5,150,5)
830 SUBEND
840 SUB RO(X1,Y1,Z,LE,RU)
850 U=0 :: CALL PATTERN(#1,44)::
     CALL POSITION(#1,X1,Y1)::
     CALL GCHAR((X1+16)/8,(Y1+16)/8,AC)
860 IF AC<>113 THEN CALL HCHAR((X1+16)/8,(Y1+16)
    /8,113):: CALL Z(Z,A,LE,RU)
870 CALL GCHAR((X1-8)/8,(Y1+16)/8,AC):: IF AC=32
```

```
    THEN CALL AB(LE,X1,Y1,U,Z,RU)
 880 IF U=1 THEN 900
 890 FOR I=1 TO 8 STEP 2 ::
      CALL LOCATE(#1,X1-16,Y1+I+1):: NEXT I ::
      CALL SOUND(30,-7,5,150,5)
 900 SUBEND
 910 SUB RU(X1,Y1,Z,LE,RU)
 920 U=0 :: CALL PATTERN(#1,44)::
      CALL POSITION(#1,X1,Y1)::
      CALL GCHAR((X1+16)/8,(Y1+16)/8,AC):: IF X1>163
     THEN CALL AB(LE,X1,Y1,U,Z,RU)
 930 IF AC<>113 THEN CALL HCHAR((X1+16)/8,(Y1+16)
     /8,113):: CALL Z(Z,A,LE,RU)
 940 IF U=1 THEN 960
 950 FOR I=1 TO 16 STEP 4 ::
      CALL LOCATE(#1,X1+I+3,Y1+8):: NEXT I ::
      CALL SOUND(30,-7,5,150,5)
 960 SUBEND
 970 !
 980 ! SPIELBESCHREIBUNG
 990 SUB SPIEL
1000 CALL SCREEN(16):: CALL CLEAR
1010 CALL MAGNIFY(2):: CALL
      SPRITE(#6,84,2,30,50,#1,73,2,50,70,#2,66,2,70,
     90,#3,69,2,90,110)
1020 CALL SPRITE(#4,82,2,110,130,#5,84,2,130,150)
1030 DISPLAY AT(24,5):"Weiter mit FIRE"
1040 CALL KEY(1,K,S):: IF K<>18
     THEN 1010 :: CALL MAGNIFY(1)
1050 CALL DELSPRITE(ALL):: CALL CLEAR
1060 CALL SCREEN(16)
1070 PRINT "TIBERT WOHNT AUF EINER" :: PRINT
1080 PRINT "WUERFELPYRAMIDE.LEIDER HAT" :: PRINT
1090 PRINT "ER EINIGE GEGNER,DIE IHN" :: PRINT
1100 PRINT "VON SEINER PYRAMIDE STOSSEN" :: PRINT
1110 PRINT "WOLLEN.HILF IHM DABEI,JEDE" :: PRINT
1120 PRINT "WUERFELFLAECHE EINMAL ZU" :: PRINT
1130 PRINT "BETRETEN,OHNE RUNTERGE-" :: PRINT
1140 PRINT "STOSSEN ZU WERDEN." :: PRINT ,,,,
1150 PRINT "VIEL GLUECK!!" :: PRINT ,,,,
1160 PRINT "WEITER MIT FIRE"
1170 CALL KEY(1,K,S):: IF S=0 THEN 1170 :: IF K=18
     THEN CALL SCREEN(2):: CALL COLOR(5,16,2,6,16,2)
     :: SUBEXIT
1180 SUBEND
1190 !
1200 ! ABSTURZ TIBERT
1210 SUB AB(LE,X1,Y1,U,Z,RU)
1220 CALL POSITION(#1,X1,Y1):: CALL DELSPRITE(#3)
1230 LE=LE-1 :: DISPLAY AT(2,8):LE :: U,K=1
1240 IF LE=0 THEN CALL GRAPH(LE,RU,Z,A)
1250 CALL SOUND(500,110,2,210,2)
1260 FOR T=40 TO 44 :: CALL SPRITE(#2,80,16,X1-20,Y1)
     :: CALL PATTERN(#1,T):: K=K+1 :: NEXT T ::
      IF K<32 THEN 1260 ELSE CALL DELSPRITE(#2)
1270 CALL LOCATE(#1,85,97)::
      CALL SPRITE(#3,84,7,X1-13,Y1-8,5.5,-3)::
      SUBEXIT
1280 SUBEND
1290 !
1300 ! ERFOLGSMELODIE
1310 SUB ERF(U1,LE,RU,Z)
1320 U1=1 :: CALL DELSPRITE(#2,#3,#4,#5,#6)::
      CALL CLEAR :: CALL SCREEN(16):: FOR I=1 TO 2 ::
      TD=2000 :: T16=TD/16
1330 CALL SOUND(T16,659,1,262,3)::
      CALL SOUND(T16,698,3,262,5)
1340 CALL SOUND(TD/8,784,3,392,5)::
      CALL SOUND(T16,659,3,330,5)
1350 CALL SOUND(T16,698,3,330,5)::
      CALL SOUND(TD/8,784,3,392,5)
1360 FOR J=1 TO 2
1370 CALL SOUND(T16,698,1,247,3)::
      CALL SOUND(T16,247,5)
1380 CALL SOUND(T16,659,3,392,5)::
      CALL SOUND(T16,392,5)
1390 CALL SOUND(T16,587,3,349,5)::
      CALL SOUND(T16,349,5)
1400 CALL SOUND(TD/8,392,5)
1410 NEXT J :: IF I=2 THEN 1460
1420 CALL SOUND(T16,659,1,262,3)::
      CALL SOUND(T16,262,5)
1430 CALL SOUND(T16,587,3,392,5)::
      CALL SOUND(T16,392,5)
1440 CALL SOUND(T16,523,3,330,5)::
      CALL SOUND(T16,330,5)
1450 NEXT I
1460 CALL SOUND(T16,523,1,330,3)::
      CALL SOUND(T16,330,5)
1470 CALL SOUND(T16,659,3,392,5)::
      CALL SOUND(T16,392,5)
1480 CALL SOUND(T16,523,3,330,5)::
      CALL SOUND(T16,330,5)
1490 SUBEND ■
```

HOUSE OF USHER

Die brandneue, spannende Mischung aus Adventure und Actionspiel. Entdecken Sie das Geheimnis des Hauses Usher.
JS. C-64 **39.-**

BONGO
Begleiten Sie Bongo, die Supermaus, auf der Suche nach den Diamanten der Prinzessin. 6 Bilder, JS.
VC-20 (+16K) C-64 **39.-**

SPACE PILOT
Flugzeugkampf, 60 K. **Die Nr. 1 in England!**
JS, TA.
C-64 ..... **39.-**

GALAXY
In immer neuen Wellen stürzen sich Galagas auf Sie, die Sie entführen oder vernichten wollen.
JS, TA.
C-64 ..... **39.-**

STAR DEFENDER
Retten Sie Menschen vor der Entführung! Achten Sie auf Lander, Mutanten, Stargate...
JS, TA. VC-20 (+16 K) **39.-**

GRANDMASTER
Das stärkste Schachprogramm für Homecomputer. TA.
VC-20 (+8 K) **49.-** C-64 **79.-**

JS = Joystick, TA = Tastatur. Alle Spiele **100% Maschinensprache**. Lieferung auf **Kassette oder Diskette** mit **deutscher Anleitung**. Preise incl. Mehrwertsteuer zuzüglich DM 5.- Porto und Verpackung. Versand gegen Nachnahme oder Vorkasse.
Viele weitere Spitzen-Programme (Spiele, Utilities, Geschäftsprogramme) finden Sie in unserem **Farb-Katalog** 2/84 (DM 2.- Schutzgebühr)

Programmierer gesucht! Händleranfragen erwünscht!

HARDWARE
Joystick Quickshot II m. Dauerfeuer 39.- .... Paar 69.-
VC-20 32 K-RAM-Modul schaltbar 179.-
16 K-RAM-Modul (auf 32 K-RAM erweiterbar) 129.-
Zusätzlicher Steckplatz beim 16 K/32 K-Modul 20.-
Modulbox m. 3 Steckplätzen + 2 EPROM-Sockeln . 89.-
C-64 Koala-Pad Grafik-Tablett mit Diskette + deutscher Anleitung 269.-

KINGSOFT
»Play it again«
FRITZ SCHÄFER
Schnackebusch 4
5106 Roetgen
Tel. 0 24 08/83 19

PEEKen im Standard-BASIC

Im letzten Heft haben wir uns mit einigen Peeks und Pokes in Sachen

Voraussetzung für unsere diesmaligen Tricks ist ein Mini-Memory oder ein Editor/Assembler-Modul. Beide Module stellen dem TI-BASIC einige zusätzliche Befehle zur Verfügung, so CALL PEEK, CALL LOAD und die schon in der letzten Ausgabe angesprochenen CALL PEEKV und CALL POKEV. Letztere benötigen wir, um im TI-BASIC Sprites zu erschaffen.

Einige Einschränkungen sind aber doch vorhanden. Einmal ist die maximale Anzahl der Sprites auf 3 begrenzt, wollen wir nicht 'merkwürdige Gebilde' auf dem Bildschirm sehen, und zum anderen dürfen die Charaktersets 15 und 16 (ASCII Werte von 144 bis 159) nicht benutzt, sprich definiert werden. In dem Bereich des BDP-RAM's, der die Charakterdefinition enthält, liegen nämlich auch die Sprite-Bewegungs-Tabellen (das ist übrigens der Grund, warum in Extended-BASIC nur 14 Charaktersätze zur Verfügung stehen). Außerdem müssen wir alle für Sprites notwendigen Speicherstellen selber überwachen, das TI-BASIC nimmt hierauf überhaupt keinen Einfluß. Dieses kann zum Beispiel bei einem "BREAK" zu 'interessanten' Bildschirmdarstellungen führen.

Im VDP-RAM liegen von >0000 bis >02FF (767 dezimal) der Bildschirm (screen image table), von >0300 bis >031F die Farbinformationen für die Charaktersätze (color table) und von >0370 bis >07FF die Charakterdefinitionen (pattern descriptor table). Die Sprite-Beschreibungen (sprite attribute table) sind ab >0300 aufwärts abgelegt, und die Sprite-Bewegungstabelle (sprite motion table) befindet sich von >0780 bis >07FF. Wie wir sehen, überschneiden sich die Farbinformationen mit den Spritebeschreibungen (darum sind unter Extended BASIC die Farbinformationen von >0800 bis >081F gespeichert). Die Nutzung von Sprites ist deshalb nur dadurch möglich, weil die Zeichen auf dem Bildschirm nicht im ASCII-Format, sondern in einem Offset von >60 (dez. 96) geschrieben werden. Um also auf dem Bildschirm in der Mitte ein C darzustellen, müssen wir CALL POKEV (400,163) ausführen. Dabei errechnet sich der Wert 163 durch Addition aus dem ASCII-Wert von C mit 67 plus dem Offset von 96.

Dementsprechend sind die ersten Bytes der Farbtabelle frei (der Video-Display Prozessor arbeitet ja mit Farbinformationen für alle 256 möglichen ASCII-Zeichen, also 32 Charaktersätzen), und in diese freien Bytes können wir die Werte für 3 Sprites poken, ohne mit sonstigen Bildinformationen in Konflikt zu geraten. Unser Listing 1 zeigt die Verwirklichung.

POKErface

Na, das ist doch endlich mal ein Programm für den TI 99: voll von Pokes auch wenn sie teilweise überflüssig sind. Aber dazu gleich. Bedingt durch den 'geringen' Speicherplatz des VDP-RAM's wird im TI-BASIC der Pointer, der auf die Obergrenze des vom VDP belegten RAM's zeigt, beim Anwählen von TI-BASIC auf den kleinstmöglichen Wert gesetzt, und das ist die Obergrenze der Definitionen der vom Betriebssystem geladenen Zeichen, was wiederum dem ASCII-Wert 127 entspricht. Alle anderen Zeichen müssen wir ja erst durch CALL CHAR bestimmen. Da sich aber nun die Sprite-Bewegungstabelle oberhalb dieser Zeichendefinitionen (im Bereich der Zeichen ASCII 144 — 159) befindet, müssen wir erst einmal die entsprechenden Zeiger (Pointer) ändern. Dies geschieht in den Zeilen 120 und 140. Die Zeile 120 setzt —31946 (entsprechend >8324) herauf. Auf dieser Adresse steht der Pointer auf dem Ende der Charakter-Tabelle. Dann muß noch der Pointer auf den BASIC-Werte-Stock heraufgesetzt werden, der in >836E (bzw. —31890) zu finden ist, damit das BASIC-Programm einwandfrei arbeitet. Oder wir schreiben einfach statt Zeile 110 bis 140:

```
110 CALL CHAR (159,"0")
```

Damit wird das Zeichen 159 'definiert', und dieser Befehl setzt 'nebenbei' auch die erwähnten Pointer hoch.

Dann wird es in den Zeilen 160 bis 220 Ernst. In einer Schleife werden die Werte für 3 Sprites gePOKEt. Die Daten für den ersten Sprite befinden sich ab >0300 bzw. dez.768. Die Daten setzen sich aus jeweils 4 Bytes zusammen. Das erste Byte ist die Lage in Y-Position. Bei dieser Lage ist der Bildschirm in ein Feinraster unterteilt, in Y-Richtung sind dies 192 Reihen und in X-Richtung 256 Spalten. Im Beispiel werden

Dem einen sein PEEK ist dem anderen sein Pinguin.

Listing 1

```
100 REM SPRITES MIT MINI-
        MEMORY
110 REM TOP CHARAKTER
        TABLE SETZEN
120 CALL LOAD(-31964,8,0)
130 REM VALUE STACK
        ADRESSE
140 CALL LOAD(-31890,8,8)
150 CALL CLEAR
160 FOR X=0 TO 2
170 RANDOMIZE
180 REM POKEN SPRITES
        (Y-POS,X-POS,
        CHAR+96,FARBE-1)
190 CALL POKEV(768+X*4,
    RND*192,RND+96,138,
    1,208)
200 REM POKEN MOTION-TABLE
        (Y-GESCHW,X-GESCHW
        ,0,0)
210 CALL POKEV(1920+X*4,
    80,RND+40,0,0)
220 NEXT X
230 GOSUB 380
240 REM BEWEGUNG FREIGEBEN
250 CALL LOAD(-31878,3)
260 REM SPRITE-
        VERGROESSERUNG,
        TASTE DRUECKEN
270 CALL LOAD(-31788,225)
280 GOSUB 380
290 REM SPRITE-
        VERGROESSERUNG
300 CALL LOAD(-31788,227)
310 GOSUB 380
320 GOSUB 380
330 REM BEWEGUNG STOPPEN,
        NOTWENDIG, DA
        POINTER NICHT
        GELOESCHT WERDEN
340 CALL LOAD(-31878,0)
350 CALL LOAD(-31788,224)
360 END
370 REM ZEIT-VERZOEGERUNG
        UND TASTEN-ABFRAGE
380 FOR X=1 TO 200
390 NEXT X
400 CALL KEY(0,K,M)
410 IF M<1 THEN 400
420 RETURN
```

Listing 2

```
100 CALL CLEAR
110 PRINT "DAS IST EINE
    DEMONSTRATION":"WIE
    EINFACH DER
    BILDSCHIRM":
    "ABGESCHALTET WERDEN
    KANN": :
120 PRINT "DRUECKE JETZT
    NACHEINANDER":"ZWEI
    BELIEBIGE TASTEN": :
     : : : :
130 CALL LOAD(-31788,160)
140 GOSUB 200
150 PRINT "DRUECKE EINE
    WEITERE TASTE":"UM DAS
    PROGRAMM ZU BEENDEN"
160 CALL LOAD(-31788,224)
170 GOSUB 200
180 GOSUB 200
190 END
200 CALL KEY(0,K,M)
210 IF M<1 THEN 200
220 RETURN
```

diese Werte mittels RND zufällig erzeugt. Das zweite Byte ist die Position in X-Richtung. Das dritte Byte bestimmt den Charakter, im Beispiel ist dies "*" entsprechend ASCII 42 plus dem Offset von 96! Das vierte Byte bestimmt die Farbe. Hier werden die BASIC-Farbcodes genommen und davon 1 subtrahiert, 0 ist also Transparent. Richtig, da steht dann noch ein fünfter Wert. Diese 208 ist ein Trick, um weitere Sprites zu unterbinden, denn sonst würde ja die ganze Farbtabelle als Sprite abgebildet. Poken wir also 208 als fünftes Byte, so ist dies die Y-Position des nächsten Sprites, die sich unterhalb des Bildschirmrandes befindet. Diese Y-Position von 208 veranlaßt den VDP, keine weiteren Sprites mehr abzubilden.

Vier Byte pro Sprite

In Zeile 210 werden dann die Bewegungsdaten gePOKEt. Für den ersten Sprite liegen diese ab >0780 bzw. dez. 1920. Auch hier werden pro Sprite wieder 4 Bytes benötigt. Das erste Byte beinhaltet die Geschwindigkeit in Y-Richtung und das zweite in X-Richtung. Die beiden übrigens Bytes werden nur intern benötigt; belassen wir sie 0.

Lassen wir das Programm soweit ablaufen (wobei es über Zeile 230 und das Unterprogramm solange wartet, bis eine Taste gedrückt wird) bewegt sich aber trotz der eingegebenen Bewegungstabellen noch nichts. Wir müssen nun nach den vom Betriebssystem abgefragten Zeiger, wieviel Sprites in Bewegung sind, setzen. Dies geschieht in Zeile 250. Diese Speicherstelle hatten wir uns aber das letzte Mal schon näher angesehen.

Das Extended BASIC bietet nun noch die Möglichkeit, Sprites zu vergrößern. Dies ist selbstverständlich auch bei unseren Sprites möglich. Wir nutzen eine Besonderheit. In >8304 (dez. —31788) liegt eine Kopie des Wertes im VDP-Register 1 (Der VDP hat insgesamt 8 verschiedene Register, die nur beschrieben werden können, und in denen die Informationen für die Betriebsart des VDP gespeichert sind). Diese Kopie wird bei jedem Tastendruck neu in das VDP-Register 1 geschrieben. Da in diesem VDP-Register auch die Informationen über Sprite-Vergrößerung gespeichert sind, können wir in Zeile 270 >83D4 mit einem passenden Wert laden, und beim nächsten Tastendruck (Unterprogramm) wird dieser Wert dann in das VDP-Register 1 geschrieben. Der Wert 225 entspricht der einfachen Spritevergrößerung in Extended BASIC. In Zeile 300 laden wir dann die Spritevergrößerung 4, der Sprite ist jetzt vergrößert und besteht aus 4 Charakterdefinitionen. Wirksam wird auch diese Änderung wieder nach einem Tastendruck. Zum Ende des Programms müssen wir aber die beiden Sprite-Pointer wieder zurücksetzen (Zeile 340 und 350).

Die Speicherstelle >83D4 läßt auch noch ein einfaches Abschalten des Bildschirms zu (gilt auch unter Extended BASIC). Siehe Listing 2.

Auch hier wird das Abschalten (Zeile 130) und das Wiedereinschalten (Zeile 160) jeweils erst nach einem Tastendruck (Zeile 140 und 180 bzw. Unterprogramm) wirksam. Aber Achtung, auf >83D4 bzw. dez. —31788 dürfen im TI-BASIC und im Extended BASIC keine anderen Werte als 160 (Bildschirm abgeschaltet), 224 (Normalwert), 225 (Spritevergrößerung 2), 226 (Spritevergrößerung 3) und 227 (Spritevergrößerung 4) gePOKEt werden, sonst kommt das Betriebssystem oder das BASIC ganz schön außer Tritt! Die Wirkung eines Pokes auf >83D4 sehen wir jedesmal erst nach einem Tastendruck, wobei es egal ist, ob dies mit CALL KEY oder mit einem INPUT, LINPUT oder ACCEPT AT geschieht. ■

Heiner Martin

ACHTUNG!

TI-99 / 4A Besitzer

- orig. TI-Peripherie-Erweiterungsbox mit Disk-Steuerkarte u. Laufwerk 1480,- DM
- RS 232 Karte parallel/seriell (für Box) 398,- DM
- 32 K Byte RAM-Erweiterung (für Box) 425,- DM
- Centronic-Schnittstelle (ohne Box anschließbar) 319,- DM
- Extended Basic-Modul 290,- DM
- Mini Memory-Modul 290,- DM
- Editor Assembler 198,- DM
- Parsec 75,- DM

Alle Preise incl. MwSt. zuzügl. 5,- DM Versandkosten. Lieferung per Nachnahme oder Vorkasse, ab 200,- DM Versandkostenfrei.

Programm-Service REIS GmbH

D-5584 Bullay, Bergstr. 80, ☎ 06542/2715

Der kleine Picasso

Graphiktableaus sind ein Schlager auf dem Peripheriemarkt. Wir haben das Super Sketch Tableau für den TI 99/4A getestet.

Lange mußten wir auf das Tableau warten. In Anzeigen wurde es schon lange angekündigt. Jetzt wurde uns das erste verfügbare Exemplar von der Firma Radix Bürotechnik zum Test bereitgestellt.

Während andere Tableaus mit schikken Griffeln aufwarten können, hat das Super Sketch nur einen beweglichen Zeichenarm vorzuweisen, der im Tableau eingebaut ist. Weiteres Zubehör sind ein ausgesprochen dünnes Handbuch, in englischer Sprache verfaßt, und ein paar Vorlagen zum Zeichnen.

Da wir von COMPUTER Praxis uns als kreative Menschen einstufen, legten wir diese Malvorlagen beiseite.

Um das Tableau in Betrieb zu nehmen, muß nur das Modul in den Modulschacht gesteckt werden. Danach Power Schalter auf "On", Super Sketch Tableau im Menü anwählen, und schon erscheint das Eingangsbild.

Ab sofort übernimmt das Tableau die gesamte Steuerung. Ein Druck auf die Menütaste am Tableau genügt, und schon haben wir den Salat. Wir befinden uns im Zeichenmenü. Über den beweglichen Zeichenarm wird ein kleiner Pfeil im Menü auf die gewünschte Funktion gestellt, und es geschieht noch gar nichts. Ein Druck auf die Selecttaste erst wählt die Funktion aus, die dann rot unterlegt erscheint. Aber noch steht der Pfeil im Menü. Erst ein Druck auf die Lifttaste läßt den Pfeil als Cursor auf die Zeichenfläche springen. Und jetzt kann es endlich losgehen. Hat man die Drawfunktion gewählt, so wird durch Bewegen des Zeichenarms eine dünne Linie gezogen.

Eine wackelige Angelegenheit

Und schon zeigt sich eine Schwäche. Da der bewegliche Zeichenarm nur in einem Punkt gelagert ist, entstanden statt waagerechter Linien immer Kreisbögen. Durch kontinuierliches Hochschieben oder Herunterziehen des Zeichenarms läßt sich dieser Effekt ausgleichen. Das Menü bietet hier Hilfe. Eine Circlefunktion suchten wir zwar vergeblich. Ebenso fehlen die Funktionen Zoom zum Vergrößern einzelner Bildpartien oder Mirror zum Spiegeln eines achsensymetrischen Körpers. Dafür können im Untermenü "Expert" Rechtecke beliebiger Form und Größe oder die so schwierigen waagerechten und senkrechten Linien erstellt werden. Vom Expert-Untermenü wird auch die Bildspeicherung gesteuert. Die Funktionen Load und Save beziehen sich zwar nur auf einen Cassettenrecorder, funktionieren aber mit bestechender Sicherheit. Bis zum Speichern eines fertigen Bildes ist es aber noch ein weiter Weg.

Wie in Öl

Hat man sein gewünschtes Bild mit der Draw Funktion in den Umrissen auf den Bildschirm gebracht, wird sofort der Wunsch nach Farbe laut. 16 Farben stehen zur Auswahl, die auch noch mit unterschiedlich breiten Pinseln verwendet werden können. Wir haben natürlich reichlich Gebrauch davon gemacht, wie ihr an den Bildschirmfotos unschwer erkennen könnt.

Es lassen sich Effekte erzielen, als ob in Öl gemalt wurde. Eine sehr feine Rasterstruktur, die sich über beliebige Zeichenstellen legen läßt, erzeugt eine echtaussehende Leinwandstruktur. Sehr hilfreich ist hierbei die Fillfunktion, die sich mit anderen Funktionen kombinieren läßt. Einmal angewählt, füllt sie alles innerhalb eines geschlossenen Kurvenzuges aus. Ob Farbe oder Raster. Aber Vorsicht! Wenn Euer Kurvenzug irgendwo auch nur ein kleines Loch hat, läuft alles aus und euer Bild ist hin. Die Farbe und das Raster laufen über das gesamte Bild und können es bis zur Unkenntlichkeit verstümmeln.

Statt der Fillfunktion können aber die unterschiedlich breiten Pinsel zum Ausfüllen benutzt werden. Das bedeutet zwar mehr Arbeit, ist dafür aber wesentlich sicherer.

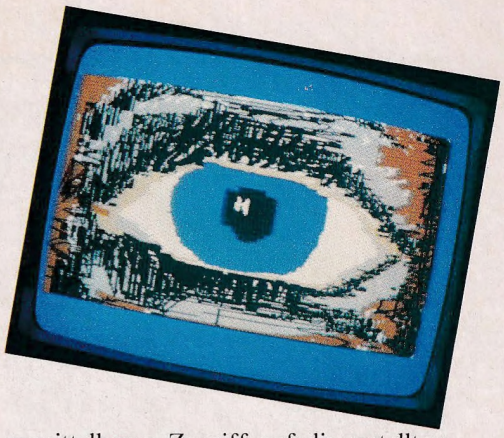
Über Modul bekommt das Super-Sketch Kontakt zum Rechner.

An den Berührungsstellen zweier farbiger Flächen kommt es manchmal zu unschönen Erscheinungen. Statt einer klaren Trennung verwischen die Farben, laufen ins andere Farbfeld. Das sieht nicht nur unschön aus, sondern ist auch sehr lästig, da eine Korrektur nötig wird.

Grenzverkehr

Werden einem Character mehrere Farben zugeordnet, reagiert der Rechner mürrisch. Statt der Farben erscheinen nur schwarze Blöcke. Wer einer roten Sonne am blauen Himmel mit einer weißen Wolke davor noch einen schwarzen Strich verpassen will, darf sich nicht wundern, wenn er nur noch schwarz sieht. Das liegt aber nicht am Tableau, sondern am TI 99/4A mit seinen beschränkten Farbmöglichkeiten.

Das Manual stellt sich als kleines,



Wir ließen einen Achtjährigen zeichnen und versuchten uns selbst.

dünnes Heftchen vor. Seltsamerweise werden aber alle Eigenschaften des Tableaus ausreichend erklärt. Selbst derjenige, der kaum der englischen Sprache mächtig ist, wird mit diesem Heftchen genauso gut klar kommen wie mit dem Tableau.

Nur eines soll hier als wirklicher Nachteil gewertet werden: Eine Möglichkeit, die Bildschirmmalereien über einen Drucker das Licht der Welt erblicken zu lassen, existiert genauso wenig wie die Möglichkeit, die Bilder über ein BASIC Programm abzurufen. Die auf Cassette abgespeicherten Bilder lassen sich zwar über OLD DS1 ohne Super Sketch Modul in den Rechner laden, aber wie und wo das Bild im Rechner steckt, darüber schweigt sich das Manual aus. Damit wird die Bildschirmmalerei zum Selbstzweck, und das finden wir von COMPUTER Praxis ausgesprochen mau! Soll doch ein so schönes Werkzeug wie das Super Sketch Graphik Tableau als Peripheriegerät den Weg in den Rechner für den Laien eröffnen und auch die Bedienung anderer Peripheriegeräte ermöglichen.

Das dicke Ende

Aber dieser Mangel soll bald behoben werden. Wie uns die Firma Radix kurz vor Redaktionsschluß mitteilte, wird jedem Käufer des Tableaus in Kürze eine Diskette nachgeliefert werden, die einen unmittelbaren Zugriff auf die erstellten Bilder ermöglicht und damit die Einbindung in andere BASIC-Programme.

Wir von COMPUTER Praxis meinen, daß das Super Sketch Graphik Tableau trotz des hohen Preises von rund 300 Mark eine lohnende Anschaffung für den TI 99/4A darstellt. Wird doch jedem User die Möglichkeit gegeben, auf einfachste Weise komplizierte Graphiken zu erstellen. Und das einzige, was man dazu braucht, ist die Konsole, ein bißchen Übung und Ausdauer.

Übrigens, die Super Sketch Graphik Tableaus kommen definitiv Ende August auf den Markt. Vielleicht werdet ihr dann alle zu kleinen Picassos! ■

K. W.

Achtung!!!

RADIX

RADIX
Bürotechnik
Handelsgesellschaft mbH
Bornstraße 4 · 2000 Hamburg 13
Tel. 040/44 16 95 · Telex 21 36 82 radix d
tägl. 10.00–12.30 + 13.30–18.30 Uhr
Sa. 10.00–13.00 Uhr

Original TI-Produkte · Fremdanbieter für TI 99/4A · Importartikel für TI 99/4A

Umfangreiches Angebot für Texas Instruments

TI 99/4A

Hardware:

TI-Box, Disklaufw., Contr. 32 K-Erw.+10 Disk.	1898,–
32 K-Erw. extern	428,–
Druckerinterface extern	348,–
Drucker GP 50 + Kabel + Druckerinterface extern	798,–
dito mit GP 550	1198,–
dito mit EPSON RX 80	1448,–
dito mit EPSON FX 80	1990,–
Graphik Tableau	298,–
Sprachsteuereinheit	348,–
RGB-Monitor+Modulator	a. A.
Superjoyst. zweifach	75,–

Mini Memory **295,–**
Terminal Emulator **199,–**
Bitte Neuheitenliste anfordern!

Software:

Editor Assembler	189,–
Extendet Basic lieferbar!	295,–
Parsec	79,–
Alpiner	79,–
Tunnels of Doom	79,–
Adventure Modul	79,–
TI-Logo II	320,–
TI-Writer	320,–
Mathematik I	99,–
Car Wars	39,–
Munch Man	79,–
Tomstone City	39,–

US-Spiele:
Moon Mine, Sewermania, Bigfoot, Meteor Belt, M*A*S*H, Microsurgion, Demon Attack, Hopper, Star Treck, Jawbreaker II, Slymoids, Munchmobile, Moonsweeper, Baseball*, Terry Turtle's Adventure*, Fathom Burgertime, Pirats-island, Bad Rogers
je Modul **99,–**
***Sprachsteuereinheit notwendig!**

Atari-Spiele TI 99:
Defender, Donkey Kong, DIG-DUG, Moon Patrol, Pac Man, Centipede, Mrs. Pac Man, Pool Position Jungle Hunt
je Modul **99,–**

Sonstiges:

Flugsimulation	49,–
Staubschutzhaube aus Kunstleder	29,–

und, und, und, und!

Graphik Tableau für TI 99 = **298,–**
für VC 64 = **298,–**

SHARP
Pocket Computer
PC 1500 375,– CE 150,– PC 1500 A, PC 1245, PC 1251, CE 125 PC 1401, CE 126 p PC 1212, CE 122
Sowie Systemhandbücher + Zubehör

SHARP MZ-700
MZ 731, Personal-Computer, 64 kB-RAM inkl. Kassettenrekorder, Farbgrafik-Drucker, eingebauter HF-Modulator und RGB-Anschluß **1190,–**
SFD 700 – Floppy für MZ-700 mit 280 KB **1390,–**

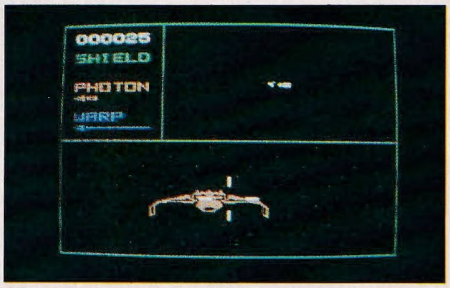
Brother, Silver Reed, Seikosha, HP, 3 M Scotch, Apple, Sanyo, Commodore, Atari

Zubehör:
Monitore, Disketten, Druckerbuffer, Disketten-Organisation, Computer-Möbel, Papier

Abdeckhauben
Kunstleder mit Leinen
für TI 99/4A, CBM 64, Atari 600, MZ 700 29,90
FX 80 39,90 QX 10 99,00
Sonderanfertigungen kein Problem!

Preise Stand 1. 6. 84 Alle Preise incl. MwSt. · Preisliste anfordern! · Lieferung erfolgt per NN oder gegen Verrechnungsscheck.
Bestellungen über DM 500,– werden frei Haus geliefert, unter DM 500,– werden DM 5,– Versandpauschale berechnet.
RADIX Bürotechnik · Bornstraße 4 + Heinrich-Barth-Straße 13 · 2000 Hamburg 13 · Telefon 040/44 16 95 · Telex 21 36 82 RADIX

SOFTWARE IM TEST



Star-Trek

Wer kennt sie nicht, die Helden des Raumschiffes "Enterprise", Captain Kirk und seine Handlanger, ständig auf Achse, um die Bösen des Weltalls, die Klingonen, zu bekämpfen?

Auch der Kämpfer am heimischen Computer kann den großen Kirk jetzt in seiner schwierigen Mission unterstützen, gleich, ob er einen C-64, einen Atari oder einen TI 99/4A sein eigen nennt.

Für diese Computer-Typen haben wir das Spiel "Star-Trek" gründlich in die Mangel genommen.

Ganz ehrlich, unser TI-Spezi war schon recht erstaunt, was 'sein' Computer alles kann! Da findet er auf dem Bildschirm ein Langstrecken-Radar, ein Sichtfenster mit Fadenkreuz und die üblichen Anzeigen für Schutzschild, Photonen-Raketen und Warp-Energie, die ein vielfaches der Lichtgeschwindigkeit ermöglicht.

"Üblich?" Üblich ist eine derart luxuriöse Ausstattung für den TI nun doch nicht, denn das Sichtfenster ermöglicht einen 3-D-Ausblick, die Feindschiffe heißen nicht nur Klingon-Kreuzer, sie sehen auch so aus.

Beim TI läßt das 'S' die Enterprise nach Backbord drehen, das 'D' nach Steuerbord, 'K' schaltet das Impuls-, 'H' das Warp-Triebwerk.

Da die Klingonen vernichtet werden müssen, ist es wichtig zu schießen. Womit? Mit Phaser und Photon-Torpedo, für die man die Tasten 'J', 'L' und 'Y' gebraucht.

Die Klingonen schießen jedoch zurück. Das Schutzschild schaltet zwar automatisch, kann aber nur 5 Treffer verkraften. Man hat auch nur 5 Photon-Torpedos an Bord und 5 Energie-Einheiten für das Warp-Triebwerk.

Der Kämpfer muß also höllisch aufpassen, daß er keine Dresche bezieht. Geht ihm dennoch die Energie für die Schilde aus, kann er auf der nächsten Sternen-Basis auftanken.

Die Steuerung mit dem Joystick ist da schon etwas komplizierter:

Links-Drehen und Rechts-Drehen ist klar, ebenso Impulstriebwerk und Phaser, der auf den Feuerknopf reagiert. Aber wie kommt man an seine Photon-Torpedos heran, wie aktiviert man das Warp-Triebwerk? Man kann natürlich die entsprechenden Tasten 'L' und 'K' der Tastatur benutzen, aber die Pro-

Einstieg zum TI

Für den Einsteiger hat Sybex »spielen, lernen, arbeiten mit dem TI-99/4A« auf Lager, ein Buch, das auch dem technisch Ungebildeten das Beherrschen des Programmierens ermöglichen kann.

Bereits mit der Minimalkonfiguration Grundgerät und Fernseher können die meisten Programme nachvollzogen werden.

Empfohlen wird jedoch zusätzlich als Basisausstattung für vernünftiges Arbeiten: Cassettenrecorder und passendes Anschlußkabel, Extended Basic-Modul und das Programmierhandbuch TI BASIC/Extended BASIC vom TI Learning Center (COMPUTER Praxis hat es in Heft 4/84 besprochen). Programme, die ohne Diskette oder Drucker nicht laufen, werden nicht behandelt.

Erklärtes Ziel der Autoren ist, uns Lesern einen spielerisch angenehmen Weg in die Benutzung unseres Computers aufzuzeigen. Und so finden wir in der Toolbox Grundlagen wie

spielen, lernen, arbeiten mit dem TI-99/4A
Schmidt/Raabe
Sybex
ISBN 3-88745-039-6
192 Seiten 28,- DM
ThoK

das Verzögern des Programmablaufs, Gestaltung eigener Zeichen, Umgang mit Farben, Sound und der hochauflösenden Graphik des Gerätes. Wir finden die 'programmierbare TI-Orgel', Kalender-Berechnungen, Biorhytmus und Balkendiagramm. Die 'Programme für private Anwendungen' stammen aus den Bereichen Prozent- und Zinsrechnung, Tabellendarstellung, Kalkulation und Investitionsrechnung.

Ein Anhang mit ASCII-Tabelle, Unterlagen zur Musterherstellung, Verzeichnis des Maschinen-Codes des TMS 9900, der Speicherbelegung der CPU und einem Extended-Basic Befehlsverzeichnis rundet den guten Eindruck, den wir gewinnen konnten, ebenso ab wie der Umstand, daß alle Listings sehr gut lesbar sind.

Allein — die Spiele sind zu kurz gekommen. Doch der Einsteiger, der seine Spiele-Listings aus anderen Quellen besorgt, ist mit diesem Buch gut bedient.

grammierer des Spieles haben den Joystick genutzt: Photonen-Torpedos werden verschossen, wenn der Joystick nach unten, also zum Spieler gezogen wird. Drückt man zusätzlich den Feuerknopf, so rast die Enterprise mit Warp durchs All. Eine Steuerung, die uns zu Anfang ganz schön zu schaffen gemacht hat. Übung macht den Meister.

Unterschiede in den Versionen bestehen nur in den verschiedenen graphischen und akustischen Möglichkeiten der Rechner. So ist die Graphik des Atari und des C-64 trotz aller Anstrengungen deutlich besser als bei dem TI, ähnlich verhält es sich mit dem Sound.

Dennoch braucht kein User neidisch zum anderen System zu schielen, denn Spielaufbau und Geschwindigkeit sind völlig gleich.

Der Freund des Dauerfeuers ist mal wieder um einen Krieg reicher geworden. Heftiges Geballer und ständiges Manövrieren garantieren hohe Punktzahlen und langes Überleben.

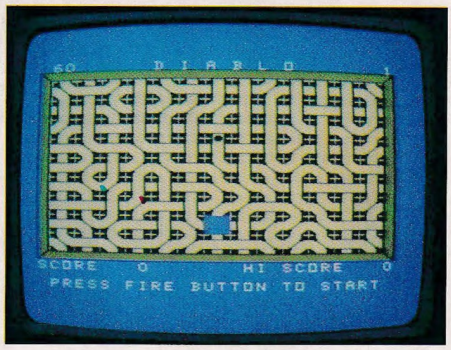
Daß die Anleitungen zum Teil nur in Englisch vorliegen, ist bei einem Spiel dieses Aufbaus nur in Bezug auf die etwas ungewöhnliche Joystick-Kontrolle von Nachteil.

Graphik:

Sound:

Spielidee:

Testergebnis:



Diablo

Die meisten TI 99/4, /4A, die noch im Dienst sind, werden in der Grundkonfiguration, das heißt Keybord mit Extended-Basic, benutzt. Entsprechend dünn ist die Auswahl der Games für den TI auf Diskette.

Ständig auf der Suche nach Neu- und Besonderheiten hat COMPUTER Praxis eine TI-Spieldiskette gefunden. Der Titel: Diablo.

Hier haben wir ein Labyrinthspiel, das hohe Konzentration erfordert. Wer schnelle Action erwartet, wird scheinbar enttäuscht. Denn das anfangs als ach so träge abgetane Spiel brachte unsere Redaktion ganz schön auf Trab, und so mancher Mitarbeiter traute sich gar nicht erst ran an den Feind!

Der Feind ist eine kleine, schwarze Kugel, die in einer Rinne läuft. Das Dumme ist, diese Rinne ist häufig unterbrochen. Fällt die Kugel aber heraus, war's das mal wieder. Es gilt also durch geschicktes Verschieben den Weg der Kugel zu ebnen, genauer: fortzusetzen. Was zunächst noch wie eine 'leichte Übung' aussieht, wird schnell zur schweißtreibenden Schlepperei, denn das benötigte Element zur Verlängerung der Rinne liegt natürlich immer genau da auf dem Bildschirm, wo es nicht gebraucht wird. Überlegtes Verschieben der Elemente garantiert also ein langes und interessantes Spiel.

Das teuflische Diablo bietet die Bedienung über Tastatur oder mittels Joystick an und ist, bei genauerem Hinsehen, in TI-XBASIC geschrieben. Trotzdem ist es nicht das übliche TI-Dünngraphik- und Schleich-Action-BASIC-Spiel. Dafür ist schon dadurch gesorgt, daß für die beiden Versionen, Tastatur und Joystick, verschiedene Programme geladen werden. Die Graphik, nur auf den ersten Blick verwirrend, nutzt alle Möglichkeiten des Rechners.

Sound finden wir nur im Wahl-Menü und als Spiel-Einleitung. Die Programmierer scheinen ganz bewußt auf verschleppende Elemente verzichtet zu haben. Sound ist so ein Element.

Die Spielidee ist wohl eine der originellsten, die für den TI umgesetzt wurden. Ein Spiel, in dem endlich mal nicht geballert wird.

Wollen wir hoffen, daß mehr aus dieser Richtung, nicht nur für den TI, auf dem Markt erscheint.

Graphik:

Sound:

Spielidee:

Testergebnis:

Der Sound aus dem die Spiele sind

Wir vergleichen, was Systeme drauf haben

Wie soll man eine schöne Jungfrau retten, wenn man ihren Hilfeschrei nicht hört? Wie soll man sein Raumschiff schützen, wenn nirgendwo eine Alarmsirene ertönt? Wie kann man sich vor einem Drachen fürchten, wenn lediglich eine jämmerliche Gestalt lautlos ein paar rote Kleckse auf den Bildschirm spuckt, die "Feuerfauchen" sein sollen?

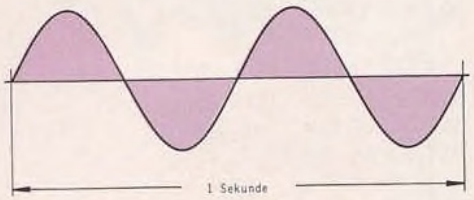
Nur Geräusche machen Illusionen perfekt. Auch beim Programmieren gehört Klappern zum Handwerk.

Aber wie will ein soundhungriger Computerneuling aus Hunderten von Frequenzen perfekte Effekte zusammenbasteln, wenn ihm z.B. im TI-Handbuch mitgeteilt wird, daß eine bestimmte Verzerrung "weißes Rauschen" erzeugt und eine andere "periodisches Rauschen"? (Kleiner Tip für zukünftige Anleitungsbuch-Schreiber: Nicht alle Computerfreaks haben die Bochumer Musikhochschule besucht). Auch das Commodore-Handbuch plaudert locker von "Hüllkurven", "Klangfarben" oder "Hochtonfilter". Da macht's richtig Spaß, Explosionen und Pistolenschüsse zu programmieren. (Welche "Klangfarbe" hat denn ein Pistolenschuß)?

Bevor es an einige grundsätzliche Prinzipien der Sound-Programmierung geht, hier schnell noch ein paar Trockenübungen, die für das Verständnis der Programm-Beispiele nützlich sind.

▷ Frequenz: So wie ein Gummiring sich in Schwingung versetzt und Töne von sich gibt, sobald man an ihm zupft, so hat jeder Ton

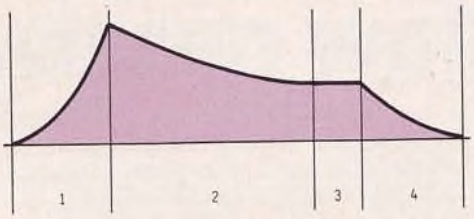
seine eigenen Schwingungen. Eine Frequenz ist nichts anderes, als die Anzahl der Schwingungen eines Tons. Als Maßeinheit wird Hertz (Hz) genommen; das bedeutet Schwingungen pro Sekunde. Unsere Abbildung zeigt eine Schwingung von 2 Hertz.



▷ Weißes Rauschen: Wenn man alle Spektralfarben auf einen Punkt bündelt, erhält man weißes Licht. Ebenso wird von einem "Weißem Rauschen" gesprochen, wenn aus allen verfügbaren Frequenzen zufällig gewählte Schwingungen zusammenklingen. Weißes Rauschen klingt unrythmisch, knurrend und knatternd und läßt sich daher für alle Explosions- und Donnergeräusche hervorragend einsetzen.

▷ Periodisches Rauschen: Hier ertönen zufällige Frequenzen in zeitlich gleichen Teilabschnitten. Das wirkt oft rhythmisch, summend und surrend und läßt sich daher gut für Maschinen- und Actiongeräusche einsetzen. Weltmeister im periodischen Rauschen ist Atari, der für jeden Tonkanal über drei Poly-Counter verfügt (wird später erklärt), mit denen man insgesamt rund 800.000 rhythmische Effekte erzeugen kann.

▷ Hüllkurve: Das hat nichts mit Marilyn Monroe zu tun, sondern beschreibt das gesamte Erscheinungsbild eines Klangs. Die Abbildung zeigt, wie ein Ton in vier Phasen aufgeteilt werden kann.



1. Anschlag; 2. Primäre Ausschwingungszeit; 3. Dauerpegel; 4. Sekundäre Ausschwingungszeit.

Beim Commodore lassen sich alle vier Elemente eines Tons programmieren. Das erlaubt die Nachahmung verschiedener Musikinstrumente.

▷ Klangfarbe: Ein Apfel aus Amerika schmeckt anders als ein Apfel aus Spanien; trotzdem handelt es sich in beiden Fällen um Äpfel. Der Ton C klingt auf einem Klavier anders als auf einer Geige, obwohl es sich um das gleiche C handelt. Klangfarben sind die unterschiedlichen Charaktere von Tonereignissen, je nachdem, was mitschwingt.

▷ Hochtonfilter: So wie ein Kaffeefilter nur die feinen Kaffeebestandteile hindurchläßt, so gelangen durch einen Hochtonfilter nur die hohen Frequenzteile eines Klangs zum Output. Neben dem Hochtonfilter gibt es noch Tiefton- und Bandpaßfilter (für mittleres Frequenz-Spektrum). Sowohl bei Atari als auch bei Commodore lassen sich Filter einschalten. Das führt zu sonderbaren, verzerrten Sounds, die z.B. in Action- und Sportspielen eingesetzt werden können.

Für Atari

Falls Sie schon mal mit den Frequenzwerten aus der Bedienungsanleitung des Atari arbeiten wollten, werden Sie festgestellt haben, daß die einzelnen Töne miteinander unsauber klingen. Der Grund dafür ist ganz einfach: Die Tabelle stimmt nicht. (Das haben die Leute von Atari sicher gemacht, um zu sehen, ob's einer merkt.)

Bei Atari werden — im Gegensatz zu Commodore und TI — keine Originalfrequenzen eingegeben, sondern in BASIC-Werte umgerechnete Frequenzen. Die Formel dazu lautet:

BASIC-Wert
= 31960 (Originalfrequenz + 1)

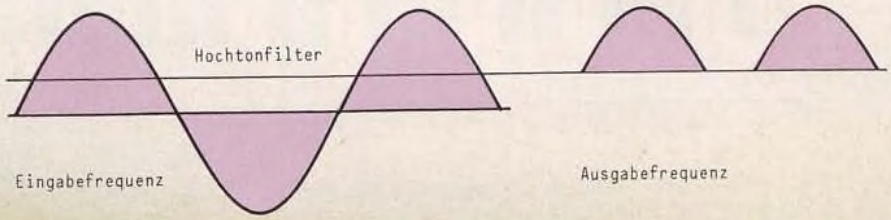
Das mittlere C z.B. schwingt mit einer Originalfrequenz von 261,62 Hz. Demnach errechnet sich der BASIC-Frequenzwert hierfür:

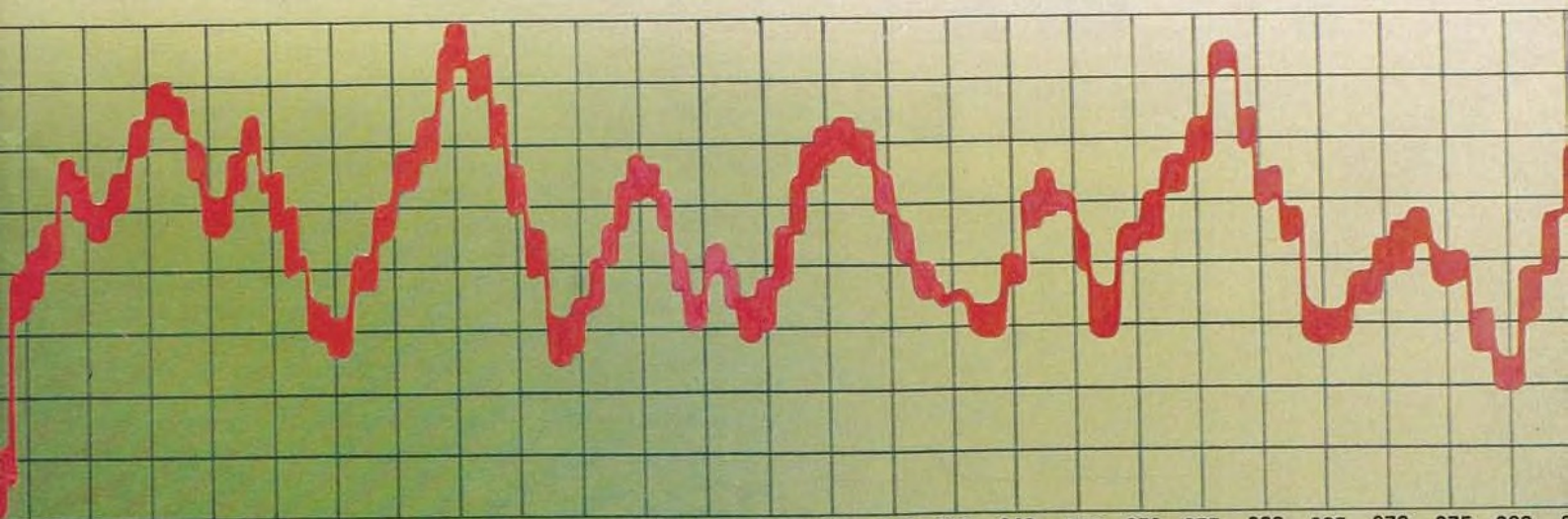
31960 (261,62 + 1) = 121,69

Aufgerundet = 122. In der Anleitung steht jedoch für das mittlere C der Wert 126. Wir bringen eine "richtig" errechnete Notentabelle auf der nächsten Seite.

Natürlich könnte man noch weitere Töne anhängen; aber ab der BASIC-Frequenz 30 klingen die Noten wie Mitternachtsgesang einer Grille mit Stimmbruch.

Außer reinen Tönen lassen sich beim Atari noch sechs verschiedene Verzerrungen einstellen, die ganz unterschiedliche Effekte bringen und für die Spiele-Programmierung besonders interessant sind.





Der Sound aus dem die Spiele sind

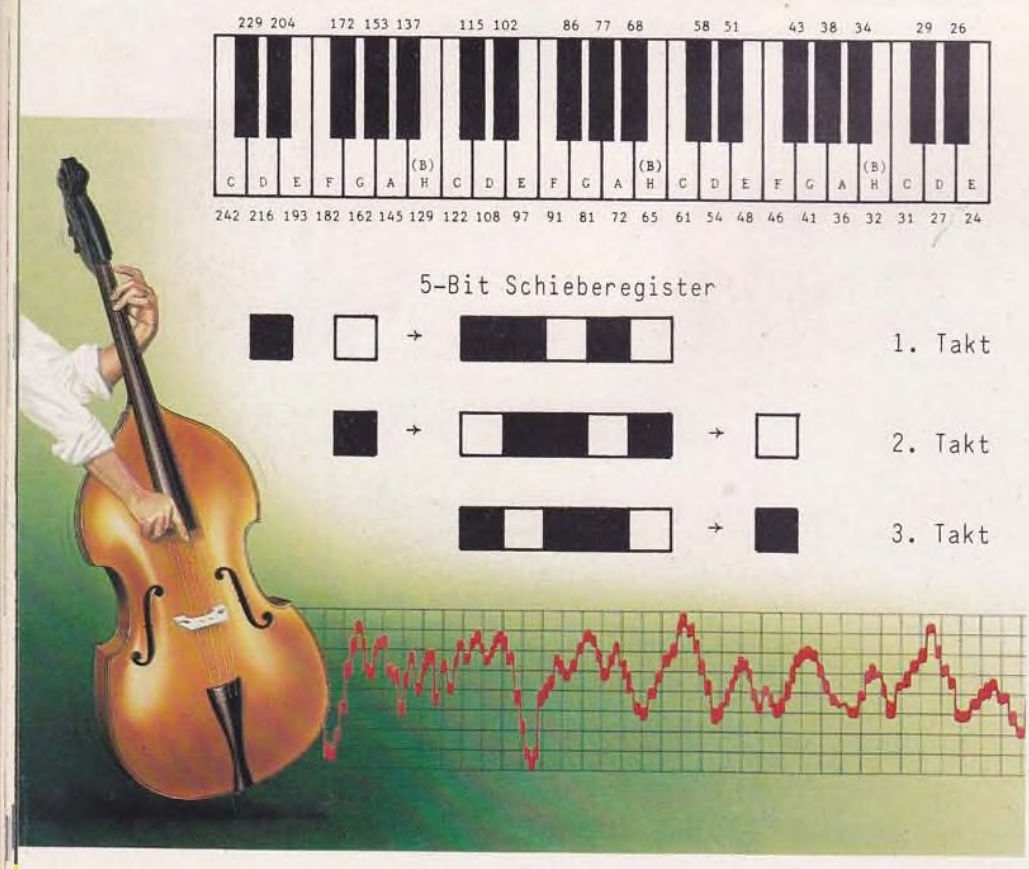
▷ Verzerrung 0: unrhythmisches, knurrendes Rauschen
Anwendung: Explosion, Schuß, Rauschgeräusche
▷ Verzerrung 2: verzerrtes, aber rhythmisch-periodisches Rauschen
Anwendung: Begleitung für Action, Bewegung
▷ Verzerrung 4: verzerrtes, knatterndes Schlagen
Anwendung: Motor, utopische Flug- und Fahrgeräusche
▷ Verzerrung 6: rythmisches Surren
Anwendung: Maschinengeräusche, Roboterschritte, Rennwagen,
▷ Verzerrung 8: unrhytmisches Rauschen
Anwendung: Explosion, Schuß, Drachen, alle Fauchgeräusche
▷ Verzerrung 12: vom verzerrten Brummen bis zum melodischen Summen
Anwendung: sehr vielseitig, für Action, Flipper, Elektronik-Geräusche

Die Verzerrungs-Angaben 10 und 14 bringen reine Töne. Bei der Programmierung von Soundeffekten mit Verzerrungs-Angaben kann es vorkommen, daß hin und wieder gar nichts ertönt. Der Grund dafür liegt in der Funktionsweise der AUDC-Register. Die Frequenzen werden nicht direkt zum Lautsprecher geschickt, sondern gelangen zunächst in einen Poly-Counter (Mehrfach-Zähler), der mit drei verschiedenen Schieberegistern verbunden ist.

Die Schwingungen, die durch den SOUND-Befehl (oder Direktzugriff) gewählt wurden, werden Bit für Bit in die Schieberegister eingeschoben und mit jedem Systemtakt des Rechners (bei Atari 1,79 MHz) um ein Register weitergeschoben. Damit allein kann natürlich noch keine Verzerrung zustande gebracht werden. Die entsteht erst dadurch, daß jedem Schieberegister eine sog. NOR-Schaltung angeschlossen ist, die sich einzelne Bits herauspflückt, in bestimmter Weise bearbeitet und wieder in das Schieberegister zurückschickt.

Es gibt für jeden Tonkanal ein 4-Bit, 5-Bit und 17-Bit Schieberegister. Je nach gewählter Verzerrung arbeiten sie in unterschiedlicher Weise und anderen Kombinationen. Dadurch entstehen die Sound-Charaktere.

Bei dieser elektronischen Schwerstarbeit kann es vorkommen, daß sich einige Frequenzwerte beim IN- und OUTPUT der NOR-Schaltung gegenseitig negieren (wahrheitsumkehren) und am Lautsprecher "Null" ankommt. Etwas einfacher ausgedrückt: Der Computer hält's Maul. Wenn das in Ihrem Programm einmal vorkommt, wählen Sie einfach eine andere Frequenz; z.B. 246 statt 247. Meistens läßt sich dadurch das Problem sehr schnell beheben.



Hier noch einige Beispiele für einfache, aber wirkungsvolle Soundeffekte auf dem Atari:

```
10 REM ATARI-BEISPIEL: SCHUSS
20 FOR X=15 TO 0 STEP -1
30 SOUND 0,10,8,X
40 FOR Y=0 TO 6:NEXT Y
50 NEXT X
```

```
10 REM ATARI-BEISPIEL:
20 REM GEFAHREN-OBJEKT
30 REM ORIGINAL RADIX
   10-LISTING
40 FOR X=-15 TO 15
50 FOR Y=200 TO 230 STEP 2
60 SOUND 0,Y,10,15-ABS(X)
70 NEXT Y: NEXT X
```

```
10 REM ATARI-BEISPIEL:
20 REM VIDEOSPIEL-SOUND
30 FOR X=16 TO 0 STEP -2
40 SOUND 0,X,6,15
50 FOR Y=0 TO 25:NEXT Y
60 NEXT X
```

Für Commodore 64

Beim Commodore ist natürlich alles anders. Zunächst gibt es eine gute und eine schlechte Nachricht. Die gute: Die Sound- und Musikmöglichkeiten sind wesentlich umfangreicher als bei anderen Modellen. Die schlechte Nachricht: Das Programmieren dieser tollen Möglichkeiten ist so kompliziert, daß der Traum vom schnellen Sound im Handumdreh'n bald ausgeträumt ist und die User lieber Omas Wellensittich zum Trillern von Soundeffekten engagieren.

Nicht weniger als sechs POKEs sind nötig, um den Widerspenstigen zum Singen zu bringen. Wenn man das jedoch erst einmal geschafft hat, kann man sich freuen. Die Klangqualität ist Spitze. Grund dafür ist ein eigens entwickelter Sound-Chip, das sog. SID (Sound Interface Device).

Die sechs Angaben zur Erzeugung eines Tons sind: Frequenzwert Low-Byte, Frequenzwert High-Byte, Wellenform bzw. Rauschen (falls gewünscht), Anschlag, primäre Ausschwingzeit, Dauerpegel, sekundäre Ausschwingzeit und Lautstärke.

Damit nicht genug. Der Ton muß nach dem Erklingen wieder ausgeschaltet werden, da es bei erneutem Aufrufen sonst zur großen Stille oder anderen Fehlern kommen kann. Für ganz vernaschte Soundkatzen gibt es außerdem noch die Möglichkeiten, verschiedene Filter, Ringmodulationen und Synchronisationseffekte zu programmieren. (Keine Panik; wird alles noch mal erklärt). Erst einmal Sounds für Normalsterbliche. Die Adressen für die Angaben von Low-Byte- und High-Byte-Frequenzwerten lauten:

Der Sound aus dem die Spiele sind

	1.Stimme	
FREQ. LOW	54272	(Basis)
FREQ. HIGH	54273	(Basis + 1)
	2. Stimme	
FREQ. LOW	54279	(Basis + 7)
FREQ. HIGH	54280	(Basis + 8)
	3. Stimme	
FREQ. LOW	54286	(Basis + 14)
FREQ. HIGH	54287	(Basis + 15)

Das Handbuch zum Commodore beginnt seine Frequenztabelle bei 16,4 Hertz. Das ist auch gut so. Denn tiefere Töne sind kaum noch wahrnehmbar; 16,4 Hertz klingt etwa so wie das Muhen einer Kuh, vorausgesetzt, es handelt sich um eine alte, müde Kuh.

Und wie funktioniert das mit dem Low- und High-Byte? — Bekanntlich kann jedes Register Werte von 0 bis 255 annehmen. Wenn Werte über 255 registriert werden sollen, müssen sie jedoch fein säuberlich auf zwei Register verteilt werden.
Die Umrechnungsformel dazu lautet:
HIGH = INT (Wert/256)
LOW = WERT — HIGH*256

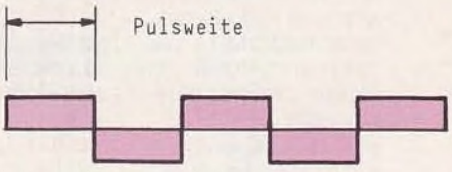
Das Low-Byte wird immer in die erste Adresse gePOKEt, das High-Byte in die zweite. Soll umgekehrt aus den beiden Dezimalwerten der Original-Wert ermittelt werden, lautet die Formel:
WERT = LOW + HIGH*256

Im Höchstfall kann also 255 + 255*256 (= 65535) auf zwei Register verteilt werden.

Da im Fall der Soundprogrammierung solche Frequenzangaben Blödsinn wären (das menschliche Ohr kann nur Töne bis etwa 14.000 Hz wahrnehmen), werden die Originalfrequenzen zunächst in eine Hilfsvariable (Frequenzparameter) umgewandelt, mit dem dann nur sinnvolle Töne von 0 bis 3952 Hz erzeugt werden können.

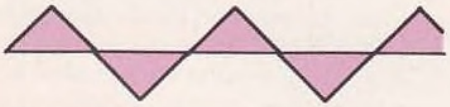
Das ist eine ganze Menge; ein normales Symphonie-Orchester bringt es bestenfalls auf 4.100 Hz (z.B. Pikkoloflöte). Die einzelnen Dezimalwerte für Low-Byte können Sie einfach der Tabelle S. 158/159 Ihres Handbuches entnehmen.

Als nächstes die Wahl der Wellenform: Man kann zwischen Puls- (oder Rechteck-) Sägezahn- und Dreieckwelle wählen.



Puls- (Rechteck) Welle

Sägezahnwelle



Dreieckwelle

Von Klangfarben war schon die Rede. Jede Klangfarbe hat ihre eigene Wellenform. Das läßt sich z.B. mit einem Oszilloskop sichtbar machen. Das können wir uns hier allerdings sparen. Schließlich übernimmt der Computer die Hauptarbeit bei der Tonerzeugung. Wichtig zu wissen ist, daß die Pulswelle einen näselnden Klang (wie Oboe), die Sägezahnwelle einen grellen Klang (wie Trompete) und die Dreieckswelle einen abgerundeten Klang (wie Blockflöte) hat.

Rauschen, Sägezahn- und Dreieckwelle können direkt programmiert werden; für die Puls- oder Rechteckwelle muß noch die Pulsbreite angegeben werden (wurde im Handbuch mit Tastverhältnis übersetzt). Die Adressen hierfür sind:

	1. Stimme			
Pulsw. LOW	54274	(Basis + 2)		
Pulsw. HIGH	54275	(Basis + 3)		
Wellenform	54276	(Basis + 4)		
Rausch	Puls	Säge	Dreieck	
129	65	33	17	←dezimal
	2. Stimme			
Pulsw. LOW	54281	(Basis + 9)		
Pulsw. HIGH	54282	(Basis + 10)		
Wellenform	54283	(Basis + 11)		
Rausch	Puls	Säge	Dreieck	
129	65	33	17	←dezimal
	3. Stimme			
Pulsw. LOW	54288	(Basis + 16)		
Pulsw. HIGH	54289	(Basis + 17)		
Wellenform	54290	(Basis + 18)		
Rausch	Puls	Säge	Dreieck	
129	65	33	17	←dezimal

Das Handbuch verschweigt leider, daß es auch möglich ist, die einzelnen Wellenformen miteinander zu kombinieren. Das bringt ganz neue Soundeffekte, die Sie ruhig mal ausprobieren können. Wenn Wellen kombiniert werden, muß den in der vorstehenden Tabelle angegebenen Werten zunächst eine Eins abgezogen werden; also 128 statt 129 usw. Erst der Gesamtsumme (z.B. 128 + 64 = 129) muß die Eins zur Aktivierung des Steuerregisters zugerechnet werden.

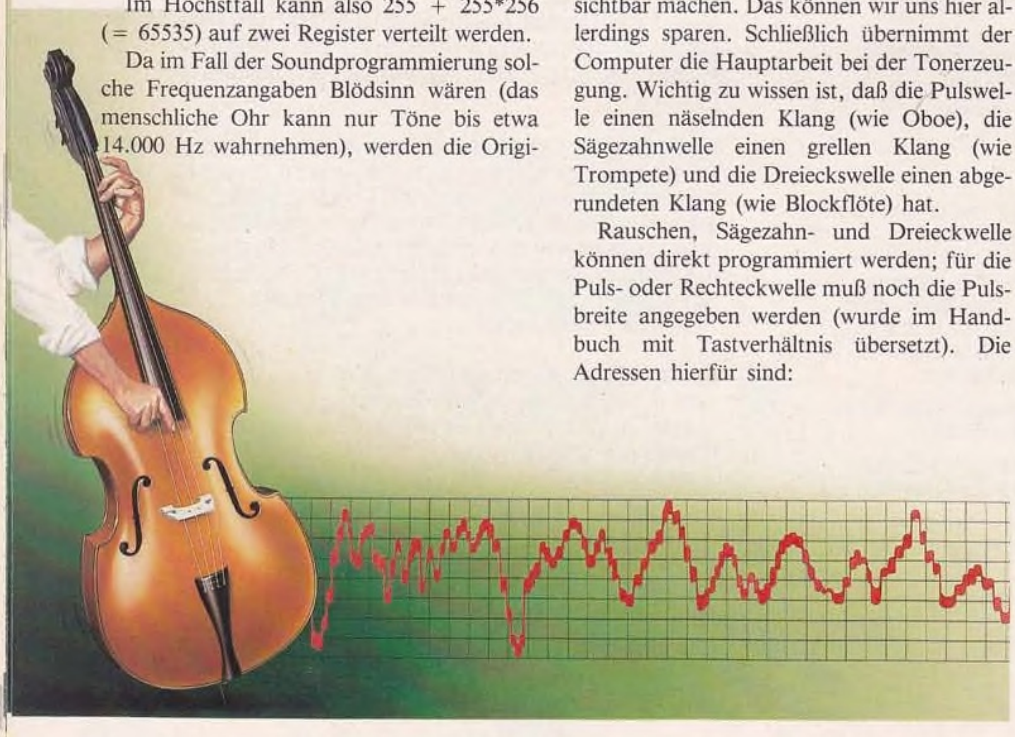
Die nächsten Angaben zur Sounderzeugung sind Anschlag/primäre Ausschwingzeit und Dauerpegel/sekundäre Ausschwingzeit. Ein Übersetzer hat dies im Handbuch mit Anschlag/Abschwellen und Halten/Ausklingen wiedergegeben. Sei's drum. Die Adressen hierfür lauten:

	1. Stimme	
An/prim. Aus	54277	(Basis + 5)
Dau/sek. Aus	54278	(Basis + 6)
	2. Stimme	
An/prim. Aus	54284	(Basis + 12)
Dau/sek. Aus	54285	(Basis + 13)
	3. Stimme	
An/prim. Aus	54291	(Basis + 19)
Dau/sek. Aus	54292	(Basis + 20)

Programmier-Beispiele für diese Adressen entnehmen Sie den nachfolgenden Listings. Die Variations-Möglichkeiten sind praktisch unbegrenzt.

Falls man einmal den längsten Anschlag und die größtmögliche Ausschwingzeit programmiert (beides 15*16), sollte man zwischendurch ruhig einen Kaffee trinken gehen. So lange braucht der Ton mindestens.

Das letzte der wichtigen Register des



Sound-Chip reguliert die Lautstärke. Die Adresse lautet 54296 (Basis + 24). Sie gilt für alle Stimmen. Es können Werte von 0 bis 15 angegeben werden. Die restlichen Bits dieses Registers sind für Tief-, Band- und Hochpaßfilter reserviert.

Hier nun einige Listings, mit denen die Programmierung der einzelnen Adressen verständlich wird:

```
10 REM COMMODORE SOUND-
   BEISPIEL 1
20 LAUT = 15:WELLE = 17:LOW = 1
30 BASIS = 54272
40 POKE BASIS + 24,LAUT
50 POKE BASIS + 6,2*16 + 5
60 POKE BASIS + 5,2*16 + 5
70 POKE BASIS + 4,WELLE
80 FOR HIGH = 100 TO
   10 STEP—0.5
90 POKE BASIS + 1,HIGH
100 POKE BASIS,LOW
110 NEXT HIGH
120 POKE BASIS + 24,0:POKE
   BASIS + 4,0
```

```
10 REM COMMODORE SOUND-
   BEISPIEL 2
20 BASIS = 54272:WELLE = 33:
   LOW = 1
30 FOR OFT = 1 TO 3:LAUT = 8
40 FOR HIGH = 60 TO 110 STEP
   2.5
50 POKE BASIS + 24,LAUT
60 POKE BASIS + 6,15*6
70 POKE BASIS + 5,0
80 POKE BASIS + 4,WELLE
90 POKE BASIS + 1,HIGH
100 POKE BASIS,LOW
110 LAUT = LAUT + 1
120 IF LAUT>14 THEN LAUT = 15
130 NEXT HIGH:NEXT OFT
140 POKE BASIS + 24,0:POKE
   BASIS + 4,0
```

```
10 REM COMMODORE SOUND-
   BEISPIEL 3
20 LAUT = 15:WELLE = 65:HIGH =
   2:LOW = 100
30 PH = 5:REM PULSWEITE HIGH
40 PL = 50:REM PULSWEITE LOW
50 BASIS = 54272
60 POKE BASIS + 24,LAUT
70 POKE BASIS + 5,5*16 + 10
80 POKE BASIS + 6,15*16 + 10
90 POKE BASIS + 4,WELLE
100 POKE BASIS + 3,PH
110 POKE BASIS + 2,PL
120 POKE BASIS + 1,HIGH
130 POKE BASIS,LOW
140 FOR Z = 0 TO 1000:NEXT Z
150 POKE BASIS + 24,0:POKE
   BASIS + 4,0
```

Für ZX-Spectrum

In der Bedienungs-Anleitung des ZX-Spectrum wird als Beispiel für die Soundprogrammierung ein Stück aus der 1. Symphonie von Gustav Mahler angeführt. Solche großen Töne kann man mit dem kleinen Radiergummitasten-Computer in Wirklichkeit nicht spucken. Er besitzt einen kleinen, eingebauten Lautsprecher und einen Beep-Generator, ähnlich wie ihn einige Weckuhren besitzen. Aber nichts für ungut.

Mit dem Spectrum lassen sich Sounds erzeugen, die man ihm zunächst einmal nicht zutraut. Der Trick ist dabei, Tondauer-Werte anzugeben, die unter Null liegen. In der Betriebsanleitung wurde leider nicht erwähnt, daß dies möglich ist. Hier einige Ergebnisse dieser Klangexperimente:

```
10 REM SPECTRUM SOUND-
   BEISPIEL 1
20 LET ZEIT = 0.0075
30 FOR T = —12 TO 24
40 BEEP ZEIT,T
50 NEXT T
```

```
10 REM SPECTRUM SOUND-
   BEISPIEL 2
20 LET ZEIT = 0.009
30 FOR T = —15 TO 15
40 BEEP ZEIT,15—ABS T
50 NEXT T
60 GOTO 30
```

```
10 REM SPECTRUM SOUND-
   BEISPIEL 3
20 FOR X = 1 TO 10
30 BEEP 0.05,1
40 BEEP 0.05,7
50 NEXT X
```

```
10 REM SPECTRUM SOUND-
   BEISPIEL 4
20 FOR X = 1 TO 5
30 READ Z,T: BEEP Z,T
40 NEXT X
50 DATA 0.1,7,0.1,4,0.1,4,0.5,9,0.6,7
```

Hüllkurvensteuerung, Wellenform oder Filter hat Sinclair nicht zu bieten. Trotzdem kann man die Sound-Trick-Kiste mit den oben gezeigten Prinzipien erweitern.

Für TI 99/4A

Ein TI-Begeisterter schrieb einmal, daß dieser Computer "über eines der umfassendsten derzeit erhältlichen Tonerzeugungssysteme verfügt."

Das darf man sicher nicht so ernst nehmen, denn der TI ist in der Soundprogrammierung relativ schwerfällig. Bei den Rauschfrequenzen wird mit den möglichen Angaben von -1 bis -8 der Eindruck erweckt, es handle sich immer um andere Verzerrungen (ähnlich wie bei Atari). In Wirklichkeit kann man jedoch nur zwischen zwei Arten (periodisches und weißes Rauschen) und unterschiedlichen Lautstärken wählen.

Damit erschöpft sich aber auch die Kritik. Zunächst einige Soundeffekte:

```
10 REM TI SOUND-BEISPIEL 1
20 FOR OFT = 1 TO 4
30 FOR F = 1500 TO 2750 STEP
   250
40 CALL SOUND (—1000,F,0,—1,0)
50 NEXT F
60 NEXT OFT
```

```
10 REM TI SOUND-BEISPIEL 2
20 CALL SOUND (90,220,0)
30 CALL SOUND (90,175,0)
40 CALL SOUND (180,—3,0)
```

```
10 REM TI SOUND-BEISPIEL 3
20 FOR F = 600 TO 200 STEP —50
30 CALL SOUND (1,F,0)
40 NEXT F
```

Bei der Musikerzeugung (also Noten übertragen, Tasten mit Orgeltönen belegen usw.) bietet der TI den Vorteil, daß Original-Frequenzwerte eingegeben werden können, und zwar von 110 bis 44733. Praktisch nutzen lassen sich dabei Angaben bis etwa 6000.

Ein praktisches Beispiel: Das Verhältnis von einer Note zur nächsten ganzen Note ist $1:\sqrt[12]{2^2}$ (=1,12246). Wenn nun eine Tonleiter gespielt werden soll, ist es nicht nötig, jede Frequenzangabe einzeln anzugeben. Es genügt, den Wert 1,12246 als Multiplikator (bei aufsteigender Tonleiter) oder als Divisor (bei absteigender Tonleiter) anzugeben. Wird z.B. der Frequenzwert 110 (Ton A) mit 1,12246 multipliziert, erhält man 123,47. Das ist die exakte Frequenz des Tons H, der in einem Ganztonschritt zu A steht. Multipliziert man nun 123,47 mit 1,12246, dann erhält man 138,59. Das ist die exakte Frequenz des Tons Cis, ein Ganztonschritt zu H.

Das Ganze funktioniert nicht nur beim Ganztonschritt. Jeder andere Notenabstand (fachmännischer ausgedrückt: Intervall) läßt sich durch die Formel $1:\sqrt[12]{2^2}$, $1:\sqrt[12]{2^3}$, $1:\sqrt[12]{2^4}$ usw. bestimmen. Das Frequenzverhältnis einer Oktave (12 Halbtonschritte) ist demnach $1:\sqrt[12]{2^{12}}$ (=2).

Mit dieser Grundlage kann man z.B. durch Tastendruck beim Orgelspiel alle gewünschten Intervalle aufrufen, ohne die einzelnen Frequenzwerte in endlos langen DATA-Zeilen speichern zu müssen. Die notwendige Angabe wird immer simultan errechnet.

Alfred Görgens

Mehr als ein Joystick

Neue Dimension für Spiele

Wie Sie am TI 99/4A einen Joystick anschließen, wissen Sie, wie ein Atari-Kompatibler angeschlossen wird, haben wir Ihnen in Heft 5/84 verraten. Hier haben wir nun einen Joystick, der an der Cassetten-Recorder-Steckdose angeschlossen wird: Das Milton-Bradley-Expansion-System, kurz MBX.

Für rund 350 Mark können Sie sich ab November das Ding zulegen. 350 Mark für einen Joystick??? Wohl doch etwas viel. Doch sollte man sich das MBX etwas genauer anschauen, bevor man derart rüde urteilt. Denn was man da für sein Geld bekommt, ist wesentlich mehr als nur ein einfacher Joystick mit den üblichen Funktionen hoch, runter, links, rechts und Feuer.

Das MBX besteht aus einer zusätzlichen Tastatur mit 64 Tasten, die am TI-Recorder- und Joystick-Port mit zwei Steckern angeschlossen wird. An dieses Keyboard werden dann ein Head-Set und ein wuchtiger Joystick angeschlossen.

Unter der Tastatur des MBX verbirgt sich nicht nur die Logik für die Expansion, sondern auch noch ein erstklassiger Sprach-Generator. Die erzeugten Stimmen sind deutlich nach männlich und weiblich unterscheidbar. Nur schade, daß die Sprach-Erzeugung andere Wege geht als die des alten TI-Speech-Synthesizers. Die alten Games, die davon unterstützt werden, hier diente uns "Parsec" als Versuchs-Kaninchen, erzeugen mit dem MBX keine Sprache.

Das Head-Set besteht aus einem Kopfhörer, ähnlich dem eines Walk-Man, an den ein Mikrophon angebaut ist. Und dieses Mikrophon ist das Besondere, denn mit ihm können die Kommandos durch Zuruf gegeben werden. In der Eröffnungs-Routine des entsprechenden Spieles macht sich der Rechner mit der Aussprache des Spielers bekannt, um dann das gesprochene Wort in die Tat umsetzen zu können. Eine riesige Unterstützung, wenn es gilt, möglichst viele Aktionen in möglichst kurzer Zeit auszuführen. Damit ist das Tastengeklimper ausgeschaltet.

Wer aber davon nicht lassen kann, der darf auf die Tastatur des MBX zurückgreifen. Eine Folie, die je nach Spiel daraufgelegt wird, zeigt, welche Taste welche spezielle Aktion auslöst.

Der Kopf ist für die Steuerung belegt, man sitzt mit leeren Händen rum und spielt?! Weit gefehlt, da ist ja noch der besagte "wuchtige Joystick".

Er liegt fantastisch sowohl in der Hand des Rechts- als auch der des Linkshänders. Der Steuerhebel auf der Oberseite hat 5 (in Worten FÜNF) Funktionen: Vorwärts, Rückwärts, Links, Rechts und Drehen. Da kommt Freude auf.

Damit nicht genug. Der Joystick ist mit vier Feuerknöpfen ausgerüstet. Was kann man da herrlich ballern...

Anders als bei einem normalen Joystick sind die Funktionen des Sticks subjektiv von der bewegten Figur aus umgesetzt d.h. "Vorwärts" bewegt die Figur in Richtung deren Nase und nicht auf dem Bildschirm nach oben. Soll die Figur sich drehen, drehe man den Knauf des Sticks.

Ein Haar hat die Suppe aber dennoch. Alte Spiele, die mit einem 'normalen' Joystick gespielt werden, können mit dem MBX nicht gesteuert werden. Aber der Aufwand, der nötig ist, die Joysticks auszutauschen, ist wohl eher als lächerlich einzustufen.

Gemeiner war da schon, daß unser Testgerät nur ein 110 Volt Netzteil hatte. Nur knapp sind wir an der Zerstörung des MBX vorbeigekommen, als wir ein Sinclair-Spectrum Netzteil, es lieferte die verlangten 9 Volt Arbeitsspannung, anschließen wollten. Die Polung des Sinclair-Netzteils ist im Stecker genau anders herum wie im Geräte-Stecker des 110Volt/9Volt-Netzteils des MBX. Das Ergebnis wäre Schrott im Wert von 350 Mark gewesen.

Wie uns der deutsche Vertrieb aber versichert, wird der Super-Joystick mit einem 220Volt/9Volt Netzteil mit korrekter Polung in den Verkauf gehen. Da sieht man mal wieder, worauf alles zu achten ist.

Für den, der gerne und viel spielt, wird dieses Gerät eine begehrte Erweiterung sein. Die nötige Software ist unterdes auch schon zu haben. Das Angebot reicht von Lern- und Computer-Anfänger-Spielen über die üblichen Wer-Schießt-Schneller- bis zu ausgefuchsten Feld-Spielen. Wir stellen Ihnen einige im Software-Teil dieses Heftes vor.

Als besonderen Leckerbissen warf uns Radix die Nachricht vor, daß einige Freaks an Befehlen arbeiten, die das MBX vom BASIC des TI aus zugänglich machen. Dann haben wir wieder den totalen Spiel-Spaß.

HU

SOGO

Der TI als ernstzunehmender Gegner in EX-BASIC

Sogo gibt's schon lange als 'hölzernes' Gesellschaftsspiel für jeden, den kniffflige Aufgaben reizen.

Die Spielregeln sind schnell erklärt: Man spielt zu zweit und setzt abwechselnd farbige Elemente auf einen von 16, zum Quadrat angeordneten Stäben. Das Ziel ist es, vier Elemente der eigenen Farbe in eine Linie zu bekommen, wobei diese auch diagonal in einer Ebene, sowie aufsteigend angeordnet sein können. Es gibt 76 Gewinnmöglichkeiten.

Zum Programm

Zuerst werden Spielernamen und die gewünschten Farben (nach TI-Codierung 1 — 16) eingegeben. Soll der Computer mitspielen, so ist als zweiter Name "TI" einzugeben. Die Spielfarbe wählt "er" sich dann selbst. Die Wahl von 1,8 oder zwei gleichen Farben wird ausgeschlossen. Nach korrekter Eingabe bleibt das Bild noch einige Sekunden stehen, ehe das Spielfeld 'aufgebaut' wird.

Spieler 1 beginnt. Die Spielelemente werden mit Hilfe der auf dem Spielfeld angegebenen Buchstaben (A — P) gesetzt. Der Buchstabe des zuletzt placierten Elements wird unter "letzter Zug" mit der entsprechenden Farbe gezeigt. Der Schriftzug SOGO erscheint in Zeile 1 des Bildschirms in der Spielfarbe von Spieler 1.

Nachdem TI seinen Zug beendet hat, erscheint wieder die Aufforderung zum Setzen. Hat irgendeiner der Spieler gewonnen, so wird dies unter Sirenenheulen angezeigt.

Um ein neues Spiel zu beginnen, muß die Leertaste gedrückt werden.

— 190: Vorbereitung der Namenseingabe
— 450: Character-Definition, Tabellendefinition und Laden.
460— 690: Spielfeldaufbau
730— 880: Programmzyklus
890—1400: Überprüfung, ob ein Zug zum Gewinn geführt hat.
1410—1490: Programmende

Spielzüge für den TI:

1540—1580: Ecken besetzen
1600—1720: Überprüfung der Gewinnchance des Gegners
1730—1990: Ist eine Zwickmühle aufgebaut?
2000—2075: Zufälliges Setzen eines Steins
2080—2180: Eigene Gewinnchance wird festgestellt
2290—ENDE: Überprüfung der gegnerischen Stellung auf Stufendiagonalen.

Variablenbeschreibung

N1$ = Namen
F1,F2 = Farben
K = Gesetzer Zug, in ASCII-Form
K1 = Zug, 1—16 für Tabelle
E = Angesprochene Ebene (1—4)
IND = Index Spieler 1 = —1
Spieler 2 = 1
CH,SP,VER,B,ZEI,Z = Variablen für Spielfeldaufbau
TID,TIS = Variable für Tabellenzugriff (TI- denkt/TI-spielt)
ZM1,ZM2 = Merker für Zwickmühlenprüfung
IA,IB,Y,ZV(x) = Variablen für ZM-Prüfung
ANZ(x) = Anzahl gesetzter Steine auf Feld x
HANZ = Höchstanzahl aller Stäbchen (... von ANZ(x))
NISE(x) = Merker für "nicht setzen"
N = Subskript für NISE
TI = Merker, ob TI spielt
ST,I,II,X = Einfache Laufvariablen.

Ronald Früchtnicht

```
100 CALL COLOR(0,5,1)
110 ON WARNING NEXT
120 OPTION BASE 1
130 GOTO 170
140 CALL CHAR :: CALL KEY :: CALL COLOR ::
    CALL SCREEN :: CALL SPRITE :: CALL DELSPRITE
150 CALL HCHAR :: CALL VCHAR :: CALL SOUND
160 F$,N1$,N2$,ZV(I),TIS,TID,J,X,Y,V,VE,I,E,N,IND,
    TI,F1,F2,F3,K,K1,K2,NR,Z,SP,CH,VER,B,ST,ZAE,ZEI,
    IA,IB,ZM1,ZM2,GC,II,NISE(I),HMERK,MERK,HANZ
170 !@P-
180 DISPLAY AT(12,2)ERASE ALL:
    "NAME 1{50*(SPACE)}NAME 2"
190 ACCEPT AT(13,10):N1$ :: ACCEPT AT(15,10):N2$
200 IF N2$="TI" THEN TI=1
210 DISPLAY AT(10,17):"FARBEN"
220 ACCEPT AT(13,20):F1
230 IF TI=1 THEN F2=16 :: DISPLAY AT(15,19):
    F2 ELSE ACCEPT AT(15,20):F2
240 IF F1>16 OR F2>16 OR F1=8 OR F2=8 OR F1=F2
    THEN DISPLAY AT(24,8):"GEHT NICHT" :: GOTO 220
250 CALL COLOR(10,F1,8,11,F2,8)
260 CALL SPRITE(#1,83,F1,1,70,#2,79,F1,1,100,#3,71,
    F1,1,130,#4,79,F1,1,160)
270 CALL CHAR(111,"3C7EF7FBFBF77E3C")
280 CALL CHAR(112,"3C7EF7FBFBFF7E3C")
290 CALL CHAR(120,"1818181818181818")
300 CALL CHAR(121,"187EFFFF")
310 CALL CHAR(122,"808080C0C0C0E0E0")
320 CALL CHAR(123,"E07070703838381C")
330 CALL CHAR(124,"1C1C0E0E0E070707")
340 CALL CHAR(125,"030303010101")
350 CALL CHAR(98,"{16*(F)}")
360 CALL CHAR(99,"{14*(0)}FF")
370 !@P+
380 DIM S(16,4),Z1(4),SP1(16),ANZ(16)
390 DIM A(10,4),DPR(20,4)
400 FOR X=1 TO 10 :: FOR Y=1 TO 4
410 READ A(X,Y)
420 NEXT Y :: NEXT X
430 DATA 1,6,11,16,4,7,10,13,1,2,3,4,5,6,7,8,9,10,
    11,12,13,14,15,16,1,5,9,13,2,6,10
440 DATA 14,3,7,11,15,4,8,12,16
450 FOR X=1 TO 20
460 FOR Y=1 TO 4
470 READ DPR(X,Y)
480 NEXT Y :: NEXT X
490 DATA 1,2,3,4,1,5,9,13,1,6,11,16,2,6,10,14,3,7,
    11,15,4,8,12,16,4,3,2,1,4,7,10,13
500 DATA 5,6,7,8,8,7,6,5,9,10,11,12,12,11,10,9,13,
    14,15,16,13,9,5,1,13,10,7,4,14,10
510 DATA 6,2,15,11,7,3,16,12,8,4,16,15,14,13,16,11,
    6,1
520 FOR I=1 TO 4 :: READ Z1(I):: NEXT I
530 DATA 13,16,19,22
540 FOR I=1 TO 16 :: READ SP1(I):: NEXT I
550 DATA 5,11,17,23,6,12,18,24,7,13,19,25,8,14,20,26
560 CALL CLEAR
570 !@P-
580 !@-@-@SPIELFELD@-@-@-@-@
590 Z=12
600 FOR SP=2 TO 5
610 Z=Z-1
620 FOR CH=122 TO 125
```

```
630 Z=Z+1
640 CALL HCHAR(Z,SP,CH)
650 NEXT CH :: NEXT SP
660 Z=12
670 FOR SP=27 TO 30
680 Z=Z-1
690 FOR CH=122 TO 125
700 Z=Z+1
710 CALL HCHAR(Z,SP,CH)
720 NEXT CH :: NEXT SP
730 CALL HCHAR(11,2,99,25):: CALL HCHAR(24,6,98,25)
740 FOR Z=9 TO 20 STEP 3
750 VER=VER+1
760 FOR SP=4 TO 24 STEP 6
770 CALL VCHAR(Z,SP+VER,120,4)
780 B=B+1 :: CALL HCHAR(Z+5,SP+VER,64+B)
790 CALL HCHAR(Z+4,SP+VER,121)
800 NEXT SP :: NEXT Z
810 !@-@-@-@-@-@-@-@-@-@-@
820 RANDOMIZE
830 IND=-1 !1.FARBE
840 CALL KEY(0,K,ST):: IF ST<>1 OR K<65 OR K>80
    THEN 840
850 IF ST<>1 OR K<65 OR K>80 THEN 840
860 FOR I=1 TO 5 :: IF NISE(I)=K
    THEN NISE(I)=32 :: DISPLAY AT(I,3):""
870 NEXT I
880 IF IND=1 THEN F3=F2 ELSE F3=F1
890 DISPLAY AT(3,8):"LETZTER ZUG :" ::
     CALL SPRITE(#6,K,F3,17,190)
900 DISPLAY AT(6,8):""
910 ZAE=ZAE+1
920 K1=K-64
930 IF IND=-1 THEN ZEI=111 ELSE ZEI=112
940 ANZ(K1)=ANZ(K1)+1 :: HANZ=MAX(HANZ,ANZ(K1))::
     IF ANZ(K1)>4 THEN 840
950 IF ANZ(K1)=4 THEN MERK=K1
960 K2=INT((K-61)/4)
970 CALL HCHAR(Z1(K2)-ANZ(K1),SP1(K1),ZEI)
980 S(K1,ANZ(K1))=IND
990 IND=-IND
1000 GOSUB 1030
1010 IF TI=1 AND IND=1 THEN 1610
1020 GOTO 840
1030 !@-@-@-@-@ PRUEF @-@-@-@
1040 !1-4 EBENE EINFACH
1050 IF HANZ>4 THEN HANZ=4
1060 FOR E=1 TO HANZ
1070 V=S(1,E)+S(6,E)+S(11,E)+S(16,E):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1080 V=S(4,E)+S(7,E)+S(10,E)+S(13,E):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1090 FOR NR=1 TO 13 STEP 4
1100 V=S(NR,E)+S(NR+1,E)+S(NR+2,E)+S(NR+3,E)::
     IF ABS(V)=4 THEN 1530 ELSE V=0
1110 NEXT NR
1120 FOR NR=1 TO 4
1130 V=S(NR,E)+S(NR+4,E)+S(NR+8,E)+S(NR+12,E)::
     IF ABS(V)=4 THEN 1530 ELSE V=0
1140 NEXT NR :: NEXT E
1150 IF HANZ<4 THEN 1520
1160 !
1170 ! 4  UEBEREINANDER ???
1180 V=S(MERK,1)+S(MERK,2)+S(MERK,3)+S(MERK,4)::
     IF ABS(V)=4 THEN GOSUB 1530 ELSE V=0
1190 ! #^#^# DIAGONAL #^#^#
1200 IF ANZ(1)<4 THEN 1240
1210 V=S(4,1)+S(3,2)+S(2,3)+S(1,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1220 V=S(16,1)+S(11,2)+S(6,3)+S(1,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1230 V=S(13,1)+S(9,2)+S(5,3)+S(1,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1240 IF ANZ(2)<4 THEN 1260
1250 V=S(14,1)+S(10,2)+S(6,3)+S(2,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1260 IF ANZ(3)<4 THEN 1280
1270 V=S(15,1)+S(11,2)+S(7,3)+S(3,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1280 IF ANZ(4)<4 THEN 1320
1290 V=S(16,1)+S(12,2)+S(8,3)+S(4,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1300 V=S(1,1)+S(2,2)+S(3,3)+S(4,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1310 V=S(13,1)+S(10,2)+S(7,3)+S(4,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1320 IF ANZ(5)<4 THEN 1340
1330 V=S(8,1)+S(7,2)+S(6,3)+S(5,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1340 IF ANZ(8)<4 THEN 1360
1350 V=S(5,1)+S(6,2)+S(7,3)+S(8,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1360 IF ANZ(9)<4 THEN 1380
1370 V=S(12,1)+S(11,2)+S(10,3)+S(9,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1380 IF ANZ(12)<4 THEN 1400
1390 V=S(9,1)+S(10,2)+S(11,3)+S(12,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1400 IF ANZ(13)<4 THEN 1440
1410 V=S(1,1)+S(5,2)+S(9,3)+S(13,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1420 V=S(4,1)+S(7,2)+S(10,3)+S(13,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1430 V=S(16,1)+S(15,2)+S(14,3)+S(13,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1440 IF ANZ(14)<4 THEN 1460
1450 V=S(2,1)+S(6,2)+S(10,3)+S(14,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1460 IF ANZ(15)<4 THEN 1480
1470 V=S(3,1)+S(7,2)+S(11,3)+S(15,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1480 IF ANZ(16)<4 THEN 1520
1490 V=S(13,1)+S(14,2)+S(15,3)+S(16,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1500 V=S(1,1)+S(6,2)+S(11,3)+S(16,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1510 V=S(4,1)+S(8,2)+S(12,3)+S(16,4):: IF ABS(V)=4
    THEN 1530 ELSE V=0
1520 RETURN
1530 !@-@-@-@- ENDE -@-@-@-@
1540 CALL SOUND(500,500,0,700,0,900,0)
1550 IF V=-4 THEN DISPLAY AT(6,8):N1$
1560 IF V=4 THEN DISPLAY AT(6,8):N2$
1570 DISPLAY AT(8,1):"HAT NACH";ZAE;"ZUEGEN GEWONNEN"
1580 CALL KEY(0,K,ST):: IF ST=0 THEN 1580
1590 CALL CLEAR :: CALL DELSPRITE(#6):: RUN
1600 !@-@-@-@-@-@-@-@-@-@-@
1610 !  TI DENKT
1620 IF S(K-64,1)+S(K-64,2)+S(K-64,3)=-3 THEN 860
1630 GOSUB 2240
1640 IF HANZ>2 THEN GOSUB 2490
1650 FOR E=1 TO HANZ
1660 TIS=1 :: IF S(1,E)+S(6,E)+S(11,E)+S(16,E)=-3
    THEN GOSUB 2150
1670 TIS=2 :: IF S(4,E)+S(7,E)+S(10,E)+S(13,E)=-3
    THEN GOSUB 2150
1680 TIS=3 :: IF S(1,E)+S(2,E)+S(3,E)+S(4,E)=-3
    THEN GOSUB 2150
1690 TIS=4 :: IF S(5,E)+S(6,E)+S(7,E)+S(8,E)=-3
    THEN GOSUB 2150
1700 TIS=5 :: IF S(9,E)+S(10,E)+S(11,E)+S(12,E)=-3
    THEN GOSUB 2150
1710 TIS=6 :: IF S(13,E)+S(14,E)+S(15,E)+S(16,E)=-3
    THEN GOSUB 2150
1720 TIS=7 :: IF S(1,E)+S(5,E)+S(9,E)+S(13,E)=-3
    THEN GOSUB 2150
```

```
1730 TIS=8 :: IF S(2,E)+S(6,E)+S(10,E)+S(14,E)=-3
     THEN GOSUB 2150
1740 TIS=9 :: IF S(3,E)+S(7,E)+S(11,E)+S(15,E)=-3
     THEN GOSUB 2150
1750 TIS=10 :: IF S(4,E)+S(8,E)+S(12,E)+S(16,E)=-3
     THEN GOSUB 2150
1760 NEXT E
1770 IF S(1,1)=-1 AND S(16,1)=0 THEN K=80 :: GOTO 860
1780 IF S(4,1)=-1 AND S(13,1)=0 THEN K=77 :: GOTO 860
1790 IF S(13,1)=-1 AND S(4,1)=0 THEN K=68 :: GOTO 860
1800 IF S(16,1)=-1 AND S(1,1)=0 THEN K=65 :: GOTO 860
1810 !@-@-@ZM/PRUEF@-@-@-@-@
1820 ZM1,ZM2,Y=0
1830 FOR E=1 TO HANZ
1840 FOR IA=1 TO 13 STEP 4
1850 IF S(IA,E)+S(IA+1,E)+S(IA+2,E)+S(IA+3,E)=-2
     THEN ZM1=1 :: GOTO 1870
1860 NEXT IA
1870 FOR IB=1 TO 4
1880 IF S(IB,E)+S(IB+4,E)+S(IB+8,E)+S(IB+12,E)=-2
     THEN ZM2=1 :: GOTO 1900
1890 NEXT IB :: NEXT E
1900 IF ZM1=0 OR ZM2=0 THEN 2060
1910 IF IA>13 THEN IA=13
1920 IF IB>4 THEN IB=4
1930 FOR I=IA TO IA+3
1940 IF S(I,E)=1 THEN 2060
1950 IF S(I,E)=0 THEN Y=Y+1 :: ZV(Y)=I
1960 NEXT I
1970 FOR I=IB TO IB+12 STEP 4
1980 IF S(I,E)=1 THEN 2060
1990 IF S(I,E)=0 THEN Y=Y+1 :: ZV(Y)=I
2000 NEXT I
2010 ZM1,ZM2,Y=0
2020 IF ZV(1)=ZV(3)THEN K=ZV(1)+64 :: GOTO 860
2030 IF ZV(1)=ZV(4)THEN K=ZV(1)+64 :: GOTO 860
2040 IF ZV(2)=ZV(3)THEN K=ZV(2)+64 :: GOTO 860
2050 IF ZV(2)=ZV(4)THEN K=ZV(2)+64 :: GOTO 860
2060 FOR I=1 TO 16 :: IF S(I,1)+S(I,2)=-2 AND ANZ(I)
     =2 AND(I<>1 AND I<>4 AND I<>13 AND I<>16)
     THEN K=I+64 :: GOTO 860
2070 NEXT I
2080 DISPLAY AT(6,12):"RND" :: K=INT(RND*16+65)::
      IF ANZ(K-64)=4 THEN 2060
2090 FOR I=1 TO 5 :: IF NISE(I)=K THEN 2060
2100 NEXT I
2110 IF ANZ(K-64)=2 AND S(K-64,1)+S(K-64,2)=2
     THEN HMERK=K
2120 GOTO 880
2130 !%*%*%*%*%*%*%*%*%*%*%
2140 BREAK :: IF E>HANZ THEN E=HANZ
2150 FOR I=1 TO 4
2160 IF S(A(TIS,I),E)=0 THEN K=A(TIS,I)+64 ::
      IF ANZ(K-64)+1<>E THEN 2180 ELSE 860
2170 NEXT I
2180 IF E-(ANZ(K-64)+1)>1 THEN RETURN
2190 FOR I=1 TO 5 :: IF NISE(I)=K THEN RETURN
2200 NEXT I
2210 N=N+1 :: IF N=6 THEN N=1
2220 NISE(N)=K
2230 FOR I=1 TO 5 :: DISPLAY AT(I,3):
     CHR$(NISE(I)):: NEXT I :: RETURN
2240 !$$$ GEW.CHANCE $$$
2250 IF HMERK=0 THEN 2270
2260 IF ANZ(HMERK-64)=3 THEN K=HMERK :: GOTO 860
2270 GC=1
2280 FOR E=1 TO HANZ
2290 IF S(1,E)+S(6,E)+S(11,E)+S(16,E)=3 THEN 2410
2300 GC=2
2310 IF S(4,E)+S(7,E)+S(10,E)+S(13,E)=3 THEN 2410
2320 FOR NR=1 TO 13 STEP 4
2330 GC=GC+1
2340 IF S(NR,E)+S(NR+1,E)+S(NR+2,E)+S(NR+3,E)=3
     THEN 2410
2350 NEXT NR
2360 FOR NR=1 TO 4
2370 GC=GC+1
2380 IF S(NR,E)+S(NR+4,E)+S(NR+8,E)+S(NR+12,E)=3
     THEN 2410
2390 NEXT NR :: GC=1 :: NEXT E
2400 RETURN
2410 FOR I=1 TO 4
2420 IF S(A(GC,I),E)<>1 THEN K=A(GC,I)+64 ::
      IF ANZ(K-64)+1<>E THEN 2450 ELSE 860
2430 NEXT I
2440 FOR I=1 TO 5 :: IF NISE(I)=K THEN RETURN
2450 NEXT I :: N=N+1 :: IF N=6 THEN N=1
2460 NISE(N)=K
2470 FOR X=1 TO 5 :: DISPLAY AT(X,3):
     CHR$(NISE(X)):: NEXT X :: RETURN
2480 !@@@STUFENDIAGONAL@@@
2490 TID=1 :: IF S(1,4)+S(2,3)+S(3,2)+S(4,1)=-3
     THEN GOSUB 2700
2500 TID=2 :: IF S(1,4)+S(5,3)+S(9,2)+S(13,1)=-3
     THEN GOSUB 2700
2510 TID=3 :: IF S(1,4)+S(6,3)+S(11,2)+S(16,1)=-3
     THEN GOSUB 2700
2520 TID=4 :: IF S(2,4)+S(6,3)+S(10,2)+S(14,1)=-3
     THEN GOSUB 2700
2530 TID=5 :: IF S(3,4)+S(7,3)+S(11,2)+S(15,1)=-3
     THEN GOSUB 2700
2540 TID=6 :: IF S(4,4)+S(8,3)+S(12,2)+S(16,1)=-3
     THEN GOSUB 2700
2550 TID=7 :: IF S(4,4)+S(3,3)+S(2,2)+S(1,1)=-3
     THEN GOSUB 2700
2560 TID=8 :: IF S(4,4)+S(7,3)+S(10,2)+S(13,1)=-3
     THEN GOSUB 2700
2570 TID=9 :: IF S(5,4)+S(6,3)+S(7,2)+S(8,1)=-3
     THEN GOSUB 2700
2580 TID=10 :: IF S(8,4)+S(7,3)+S(6,2)+S(5,1)=-3
     THEN GOSUB 2700
2590 TID=11 :: IF S(9,4)+S(10,3)+S(11,2)+S(12,1)=-3
     THEN GOSUB 2700
2600 TID=12 :: IF S(12,4)+S(11,3)+S(10,2)+S(9,1)=-3
     THEN GOSUB 2700
2610 TID=13 :: IF S(13,4)+S(14,3)+S(15,2)+S(16,1)=-3
     THEN GOSUB 2700
2620 TID=14 :: IF S(13,4)+S(9,3)+S(5,2)+S(1,1)=-3
     THEN GOSUB 2700
2630 TID=15 :: IF S(13,4)+S(10,3)+S(7,2)+S(4,1)=-3
     THEN GOSUB 2700
2640 TID=16 :: IF S(14,4)+S(10,3)+S(6,2)+S(2,1)=-3
     THEN GOSUB 2700
2650 TID=17 :: IF S(15,4)+S(11,3)+S(7,2)+S(3,1)=-3
     THEN GOSUB 2700
2660 TID=18 :: IF S(16,4)+S(12,3)+S(8,2)+S(4,1)=-3
     THEN GOSUB 2700
2670 TID=19 :: IF S(16,4)+S(15,3)+S(14,2)+S(13,1)=-3
     THEN GOSUB 2700
2680 TID=20 :: IF S(16,4)+S(11,3)+S(6,2)+S(1,1)=-3
     THEN GOSUB 2700
2690 RETURN
2700 II=5
2710 FOR I=1 TO 4
2720 II=II-1
2730 IF S(DPR(TID,I),II)=-1 THEN 2810
2740 K=DPR(TID,I)+64
2750 IF ANZ(K-64)=II-1 THEN 860
2760 FOR J=1 TO 5 :: IF NISE(J)=K THEN RETURN
2770 NEXT J
2780 N=N+1 :: IF N=6 THEN N=1
2790 NISE(N)=K
2800 FOR X=1 TO 5 :: DISPLAY AT(X,3):
     CHR$(NISE(X)):: NEXT X :: RETURN
2810 NEXT I
2820 RETURN
```

Reaktionen

Wer sich erinnert ...

In Heft 7/84 stellten wir Moonraker vor. Der Autor, Beat Schönauer, war der Meinung, man könne das Spiel zu einer Gemeinschaftsproduktion machen. Andere TI-User sollten es weiter ausbauen. Wir finden diese Idee prima.

Christoph Bladoschewski begeisterte sich auch für dieses Spiel und sandte eine heiße Ausbaustufe, die wir Ihnen nicht vorenthalten wollen. Die folgenden Programmzeilen ersetzen die entsprechenden im Ursprungslisting. 370 und 380 müssen Sie löschen.

```
  1 REM Die Zeilen 370 und 380 werden aus dem
    Listing ersatzlos gestrichen!!!
  2 REM {47*(=)}
200 ! MOONRAKER VERSION 1.1
201 ! BY  CH. BLADOSCHEWSKI
202 ! DORSTENER STR.427
203 ! 4630 BOCHUM 1
204 !
205 !     (C) 1984
206 !
230 M$="M O O N R A K E R "
240 Z=2
310 CALL LOCATE(#1,Z*8-7,N*8+49)
330 CALL LOCATE(#1,Z*8-13,N*8+45)::
     CALL MAGNIFY(2):: FOR Q=0 TO 1 ::
     CALL PATTERN(#1,Q+140):: FOR D=1 TO 4 ::
     NEXT D :: NEXT Q
340 CALL PATTERN(#1,142):: CALL HCHAR(Z,N+7,C)::
     CALL PATTERN(#1,139)
420 FOR N=5 TO 12 :: CALL COLOR(N,14,1):: NEXT N ::
     DISPLAY AT(4,6):CHR$(127)&" Version 1.0 by" ::
     DISPLAY AT(5,6):"Beat  Schoenauer"
430 DISPLAY AT(7,6):CHR$(127)&" Version 1.1 by" ::
     DISPLAY AT(8,6):"Ch.Bladoschewski" ::
     DISPLAY AT(24,2):"Beginn mit beliebiger Taste"
670 CALL CHAR(92,"{9*(0)}1077FC47F1F03{18*(0)}
    80E0FE23FEF8C")
680 CALL CHAR(33,RPT$("01",8))
690 CALL CHAR(36,RPT$("0",23)&"73F"&RPT$("0",23)
    &"F1E7CF838F87C1E0F")
760 CALL COLOR(1,4,1,2,16,1,3,16,1,4,16,1,5,16,1,6,
    16,1,7,16,1,8,16,1,9,11,1,14,16,1)
770 MOON=5 :: ZA=1 :: PUNKTE,V,RAD=0
1142 CALL COINC(#5,#9,15,C3)
1145 IF RO<3 THEN RO=RO+1 ELSE CALL DELSPRITE(#9)
1170 Z=INT(RND*10+1)
1180 ON Z GOSUB 2000,5000,2000,4000,3000,2000,3000,
    2000,4000,2000
3010 CALL POSITION(#10,X,Y):: Y1=INT(Y/8+1)::
     CALL VCHAR(6,Y1,33,14)::
     CALL SOUND(50,330,3,220,9,120,5)::
     CALL VCHAR(6,Y1,32,14)
3020 CALL SOUND(500,-7,0):: CALL LOCATE(#6,150,Y)::
     RETURN
4010 IF RO<3 THEN RETURN ELSE RO=0 ::
     PUNKTE=PUNKTE+15
4015 PO=INT(RND*256)+1
4020 ON INT(RND*4)+1 GOTO 4100,4200,4300,4400
4100 CALL SPRITE(#9,100,16,1,PO,25,0):: RETURN
4200 CALL SPRITE(#9,104,14,1,PO,25,0):: RETURN
4300 CALL SPRITE(#9,108,7,1,PO,25,0):: RETURN
4400 CALL SPRITE(#9,112,4,1,PO,25,0):: RETURN
5000 !
5010 IF RO<3 THEN RETURN ELSE RO=0 ::
     PUNKTE=PUNKTE+20
5020 CALL SPRITE(#9,36,14,130,1,0,-30):: RETURN
6000 ON ERROR 6000 :: RETURN 1090
```


Solange der Vorrat reicht + solange der Vorrat reicht + solange der Vorrat reicht +

SOFTWARE SONDERANGEBOTE

Strategie & Taktik total. Von Avalon Hill.

Pro Programm statt 139 Mark bei uns nur 69,–

Andromeda Conquest. Die ultimative Weltraum-Simulation! Händel und Handel in fernen Galaxien, Eroberung und Verteidigung von Planeten sind Voraussetzung für den Erfolg. ★ Programm (Cassette) für Atari Computer (32 K), Apple, Pet CBM, TRS 80 I/III ★ Best.Nr. AH 1 ★ **Voyager I.** Sabotage auf dem Roboter-Schiff. Im Rennen gegen die Zeit sind Sie Jäger und Gejagter zugleich. ★ Programm (Cassette) für Atari Computer (32 K), Apple II +, Pet CBM, TRS-80 Color, TRS-80 I/III ★ Best.Nr. AH 2 ★ **Guns Of Fort Defiance.** Schaffen Sie's, das Fort gegen einen übermächtigen Gegner zu verteidigen? Strategie perfekt. ★ Programm (Cassette) für Atari Computer (48 K), Appel II, Pet CBM, TRS-80 ★ Best.Nr. AH 3 ★ **Top Action von Avalon Hill: Moon Patrol.** Der rasante Arkadenknüller! High-Speed Spielspaß der Sonderklasse. ★ Programm (Cassette) für Atari Computer (16 K) und C 64 ★ Best.Nr. AH 4 ★

Die Funhouse-Fundgrube für TI-Fans.

Rabbit Trail
Rüber über den Acker und ran an die Möhren! Der Actionspaß für Fortgeschrittene. Best.Nr. TI 2

Ambulance
Na, dann karren Sie mal den total verrückten Krankenwagen durch die Stadt und ...retten Sie den Patienten! Best.Nr. TI 4

Henhouse
Der nackte Wahnsinn im Hühnerstall. Eiersammeln wie noch nie. Schneller und schlimmer geht's nimmer! Best.Nr. TI 1

Driving Demon
Hier können Sie zeigen, wie rasant Sie fahren. Lenkradbeherrschung vorausgesetzt. Und reichlich Reaktionsvermögen. Best.Nr. TI 3

Pro Programm statt 129 Mark bei uns nur 59,–

Alle TI-Programme sind Solid State Cardridges!

BESTELLCOUPON

------- Ausschneiden und einsenden -------

PORT PRESS Verlag GmbH - Papenhuder Straße 6 - 2000 Hamburg 76

Hiermit bestelle ich zu den untenstehenden Lieferbedingungen

Anzahl	Best.Nr.	Kurztitel	Preis pro Stück	Gesamt

Name _____ Vorname _____ + Versandkosten DM 5,–

Straße _____ Gesamt _____

PLZ/Ort _____

Den Gesamtbetrag von DM _____ habe ich auf Ihr Konto Commerzbank Hamburg 1810746 (BLZ 200 400 00) überwiesen/ liegt als Verrechnungsscheck anbei. (Zutreffendes bitte unterstreichen) Absender und Anschrift nicht vergessen!

CP/T 11/84

Unsere Lieferungs- und Zahlungsbedingungen: Alle hier angebotenen Programme sind nur begrenzt lieferbar. Die Auslieferung erfolgt in der Reihenfolge des Bestellungs- bzw. Zahlungseingangs. Bei Bestellung bitte deshalb unbedingt Ersatztitel angeben. Software ist vom Umtausch ausgeschlossen. **Keine Lieferung per Nachnahme.**

Eine solide Basis

Wir wollen Ihrem Computer auf die Sprünge helfen. Diesmal lassen wir ihn blinken, was das Zeug hält.

Wollen Sie BASIC lernen? Antworten Sie bitte, aber nur mit ja oder nein. So fragte uns das in der letzten Folge eingetippte Programm. Folgsam haben wir JA eingeklappert und wurden prompt zu unserer Entscheidung beglückwünscht. Ein wahrhaft berauschender Beweis künstlicher Intelligenz.

Im Detail: Zeilen 10, 11 und 20 enthalten **REM**arks. Diese Anweisungen, die eigentlich gar keine sind, werden vom BASIC-Interpreter ignoriert. Deshalb kann hinter dem REM stehen, was will: Erläuterungen zum Programm, ein Copyright-Vermerk oder Leerzeichen — es interessiert den Computer nicht!

Übersichtlichkeit und Dokumentation gehören zu den obersten Programmierertugenden, doch Vorsicht: Auch REM-Zeilen benötigen wertvollen Speicherplatz.

BASIC-Befehle im direkten Vergleich

Also kurz gesagt: Am Anfang unseres Programmes tut sich erst mal gar nichts. Doch von Zeile 30 bis 120 wird geprinted, was das Zeug hält.

PRINT (Drucke) ist eigentlich der wichtigste Befehl. Er sorgt dafür, daß auf dem Bildschirm etwas passiert. Später werden wir lernen, wie mit dem PRINT-Befehl auch andere Ausgabegeräte angesprochen werden können. Zeile 130 mit **INPUT** beschert uns ein Fragezeichen auf dem Schirm. Sinclair nutzt jedoch die beiden unteren Bildschirmzeilen. Der Computer wartet und wartet, bis wir auf der Tastatur Zeichen eingeben (die zur Kontrolle gleich auf dem Schirm erscheinen) und ihm mit **ENTER** bzw. **RETURN** das Eingetippte als komplette Dateneingabe und abgeschlossen verkaufen. Auch hier (analog zu PRINT) gilt, daß die Dateneingabe per Tastatur nur den Standardweg darstellt.

Jedenfalls: In Zeile 130 wird das, was wir als Antwort eintippen, der **Variablen** A$ zugewiesen. Wir erinnern uns: eine Variable entsteht, wenn wir bestimmte Speicherplätze unseres Rechners zu einer Gruppe zusammenfassen und dieser einen Namen, den Variablennamen, verpassen.

Was ist bei Variablennamen zu beachten? Bei einigen Rechnern dürfen sie z.B. nur zwei Zeichen lang sein. Auch ist unterschiedlich, welche Sonderzeichen außer Buchstaben und Ziffern sie enthalten dürfen. In jedem Fall aber müssen sie mit einem Buchstaben beginnen. Und: Ich muß dem Interpreter von vornherein sagen, auf welche Weise die Werte der Variable zugewiesen werden sollen.

Da gibt es zunächst die Variable, in die wir Zahlen stellen wollen: Als Name reicht ein einziger Buchstabe, z.B. A. Oder K1, K2 .... Oder WERT. Oder A1Z55. Auf jeden Fall darf der Name kein $ enthalten; denn so kennzeichne ich Variablen, in denen Zeichenketten (also Buchstaben, Ziffern und Zeichen) gespeichert werden sollen. Sie heißen **Stringvariablen**. Ich kann also bedenkenlos A$ den Wert HANS zuweisen (A$="HANS"). Wer allerdings versucht, der numerischen Variablen A den HANS zuzuweisen (A="HANS"), erfährt, was ein beleidigter Rechner ist. Entweder wird die Programmzeile gar nicht angenommen oder sie führt zu einem TYPE MISMATCH ERROR, jedenfalls: So nicht!

Also immer schön trennen. Zahlen speichern, speziell, wenn mit ihnen gerechnet werden soll, in Numerischen Variablen, Zeichenketten ausschließlich in Stringvariablen (die mit dem $ dran).

Nun weiter in unserem Programmbeispiel. In den Zeilen 150 bis 160 tauchen zwei weitere wesentliche Elemente des Programmierens auf. Mit **IF..THEN** erfolgt die Abfrage einer Bedingung. IF A$="JA" THEN. Also, wenn in der Variablen A$ der Wert JA steht, dann führe aus, was hinter THEN steht.

Das zweite Element ist **GOTO**, der Befehl, von hier zu einer bestimmten anderen Zeilennummer zu springen. Logischerweise muß die anzuspringende Zeile wirklich im Programm vorhanden sein, zumindest als REM-Zeile, damit der Interpreter weiß, wo er hin soll. Sonst fällt er auf die Nase. Insgesamt bedeutet Zeile 150 IF A$="JA" THEN GOTO 300: Wenn, während die Zeile verarbeitet

Zwar schickt er nicht gleich Laser-Strahlen, doch lassen wir den Rechner blitzen.

wird, in A$ JA steht, wird als nächstes die Zeile 300 aufgesucht und ausgeführt. Und wenn nicht? Dann wird das, was in 150 hinter THEN steht, eben nicht ausgeführt und mit der nächsten Zeile fortgefahren.

Spielen wir den Ablauf in 150 und 160 mal durch, "fahren wir mal Eisenbahn", wie es im Profi-Jargon heißt. Nehmen wir an, ein Scherzbold hätte bei der Glaubensfrage nach BASIC mit "KLAR" geantwortet. In 150 ist nichts zu tun, da die Bedingung A$="JA" nicht erfüllt ist. In 160 ähnlich: A$="NEIN" ist auch nicht. Also weiter mit der nächsten Zeile. Was tun? Da hat sich der Programmierer etwas ausgedacht. Er läßt den Rechner eine kleine Schleife abarbeiten und so eine bestimmte Zeitspanne verstreichen (damit Kalli Klaps aus Achterndiek denkt: Ha, jetzt hat's ihn gefetzt). Doch mitnichten! Der Computer fängt nochmal zu fragen an.

Doch schauen wir uns diesen Komplex im Programm nochmal an. In Zeile 200 wird der Variablen A (wohlgemerkt nicht A$) der Wert 150 zugewiesen: **LET** A=150. Bei den meisten Rechnern kann das Wort LET einfach entfallen.

In der nächsten Zeile wird von dem Wert in A 1 abgezogen, in 220 schließlich gefragt, ob A dadurch schon auf 0 reduziert wurde. Dann nämlich springt das Programm zur weiteren Verarbeitung zurück zur Eingabe.

Ist A aber noch nicht auf 0, erfolgt in 230 auf jeden Fall ein Rücksprung nach 210, wo A erneut um 1 reduziert wird. Was soll das? Nun wir wissen, daß BASIC nicht unbedingt schnell ist. Es braucht also seine Zeit, um die 150 Subtraktionen, Abfragen und Rücksprünge vorzunehmen. Und das ist genau die Zeit, die wir brauchen, um Kalli Klaps zu verblüffen.

Aktion sauberer Bildschirm

Das Ende des Programms gestaltet sich mit bekannten Mitteln. Je nachdem, welche der gültigen Antworten JA oder NEIN eingegeben wurde, werden die Zeilen ab 300 oder 400 ausgeführt. Auf jeden Fall steht in 410 der Befehl **STOP** (für TI: BREAK), der die Programmausführung unterbricht, aber zuläßt, daß mit **CONTINUE** ab der nächsten Zeile nach dem STOP weitergefahren wird.

In unserem Fall folgt ein **CLEAR**, das alle bisher vereinbarten Variablen löscht. Die Notwendigkeit, es auszuführen, schwankt von Rechnertyp zu Rechnertyp.

Mit **CLS** wird, zumindest bei SINCLAIR, der Bildschirm gelöscht, bei Commodore und Atari bringen wir zu diesem Zweck ein Steuerzeichen auf den Schirm.

Damit hätten wir das Listing der letzten Folgen erschlagen. Wir wollen jedoch versuchen, die **Verzögerungsschleife** etwas eleganter zu gestalten. Nehmen wir also die Zeilen 200—230, die 150 mal durchlaufen werden. Für solche Konstruktionen bietet BASIC die **FOR — NEXT** — Schleife, mit der bestimmt werden kann, wie häufig die darin enthaltene Statement-Folge durchlaufen wird.

Folgender einfacher Fall: Die Zahlen 1 bis 10 sollen ausgedruckt werden.

```
10 FOR K=1 TO 10
20 PRINT K
30 NEXT K
40 PRINT "DAS WAR'S"
```

Wenn die Zeile 10 das erste Mal ausgeführt wird, erhält K den Wert 1. Die Zeile 20 bringt die 1 auf den Bildschirm. Zeile 30 besagt: Hole dir jetzt für K den nächsten Wert, springe zurück nach Zeile 10. Dort erhält K den Wert 1+1, das ist nach Adam Ries 2. Dann wird wieder die 20 ausgeführt, die 2 erscheint auf dem Schirm. Und so weiter, bis in Zeile 10 aus K mit dem Wert 10 die 11 wird. Da merkt BASIC: Aha, jetzt bin ich mit der leidigen Schleife fertig, jetzt geht's nach dem NEXT weiter. Und spornstreichs wird Zeile 40 ausgeführt.

Man kann solch eine Schleife auch so programmieren, daß sie nichts ausführt als sich selber. Und so mit 2 Statements eine kleine, übersichtliche Zeitschleife programmieren, ohne GOTO (das Nirwana anspruchsvoller Programmierer ist das Listing ohne GOTO). Ersetzen wir die Zeilen 200—230 und 450—480 durch:

```
200 FOR A = 1 TO 150
210 NEXT A
220 GOTO 100
450 FOR A=1 TO 300
460 NEXT A
470 GOTO 30
```

Probieren Sie mal aus, wie sich die Veränderung des Endwertes auswirkt.

Hier einige Beispiele, wie man Zeitschleifen sinnvoll einsetzen kann.
Probieren Sie doch einfach.

Atari

```
 10 REM *BLINKY*
 20 DIM A$(20)
 30 INPUT A$
 40 PRINT CHR$(125)
 50 FOT I=1 TO 5
 60 POSITION 15,5:PRINT"HALLO"
 70 POSITION 15,10:PRINT A$
 80 FOR K=1 TO 100
 90 NEXT K
100 POSITION 15,5: PRINT
    "|9*(Space)|"
110 POSITION 15,10: PRINT
    "|20*(Space)|"
120 FOR K=1 TO 100
130 NEXT K
140 NEXT I
150 GOTO 30
```

So heißt es für Atari. Für andere Rechner müssen Sie jedoch einige Änderungen vornehmen.

SINCLAIR

```
Zeile 20 entfällt
 40 CLS
 60 PRINT AT 5,15;"HALLO"
 70 PRINT AT 10,15;A$
100 PRINT AT 5,15;"|9*(Space)|"
110 PRINT AT 10,15;"|20*(Space)|"
```

Der Sinclair kennt im Gegensatz zu den anderen Rechnern noch den Befehl PAUSE X, der eine Verzögerung bewirkt. Mit X wird deren Länge bestimmt, X=1 bedeutet 1/50 sec. Sie könnten also noch folgende Variante probieren:

```
Zeilen 90 und 130 entfallen
 80 PAUSE 50
120 PAUSE 50
```

COMMODORE

```
Zeile 20 entfällt
 30 INPUT A$
 35 IF LEN(A$)>12 THEN 30
 40 PRINT CHRS(147):D$=
    |HOME|10*DOWN|"
 60 PRINT LEFT$ (D$,6); TAB(15);
    "HALLO"
 70 PRINT LEFT$(D$,11);TAB(15);
    A$
100 PRINT LEFT$(D$,6); TAB(15);
    "|9*(Space)|"
110 PRINT LEFT$(D$,11); TAB(15);
    "|12*(Space)|"
```

Für den VC 20 sind alle Angaben TAB(15) in TAB(5) zu ändern!

TEXAS INSTRUMENTS

```
 20 B$="HALLO"
 40 CALL CLEAR
 60 FOR M=1 TO LEN(B$)
 61 CALL HCHAR(15,1+M,ASC
    (SEG$(A$,M,1)))
 62 NEXT M
 63 J=13
 70 FOR M=1 TO LEN(A$)
 71 CALL HCHAR(15,J,ASC
    (SEG$(A$,M,1)))
 72 J=J+2
 73 NEXT M
100 CALL HCHAR(15,2,32,9)
110 CALL HCHAR(15,13,32,19)
```

Überlegen Sie mal, was da wie abläuft, vielleicht zeichnen Sie mal einen PAP. Wir besprechen beim nächsten Mal dieses Programm.

Außerdem lernen Sie dann noch vieles mehr über die Aufbereitung Ihrer Daten für den Bildschirm.

ThoK

Und ewig lockt der Prozessor ...

Wieder einmal zeigt der TMS 9900, was er kann.
Mit 3-D-World können Sie so richtig in die Röhre gucken

Das menschliche Auge, ein hochspezialisiertes Organ, wird oft genug vom Computer gebeutelt und beleidigt. Der User gähnt, langweilt sich. Denn eine platte Graphik jagt die andere. Selbst so schöne Pakete, wie wir sie auch schon von Apesoft und Herrn Kull vorstellten, verlieren irgendwann ihren Reiz. Der Wunsch, dieser Monotonie ein schnelles Ende zu bereiten, wird laut. Platte Bilder, die kann mittlerweile jeder auf den Bildschirm bringen. Etwas Neues wird gesucht, etwas, das nicht jeder hat. Es sollte aber doch etwas für die ach so gestreßten Augen sein, also wieder Graphik. Und schnell muß es auch noch sein. Also Assembler. Toll wäre es dann noch, wenn die gesuchte Graphik dreidimensional wäre. Aber so etwas gibt es ja nur für größere Systeme, hört man klagen. Gab es bisher nur für größere Systeme! Unsere Antwort lautet ganz schlicht: Hega-Soft.

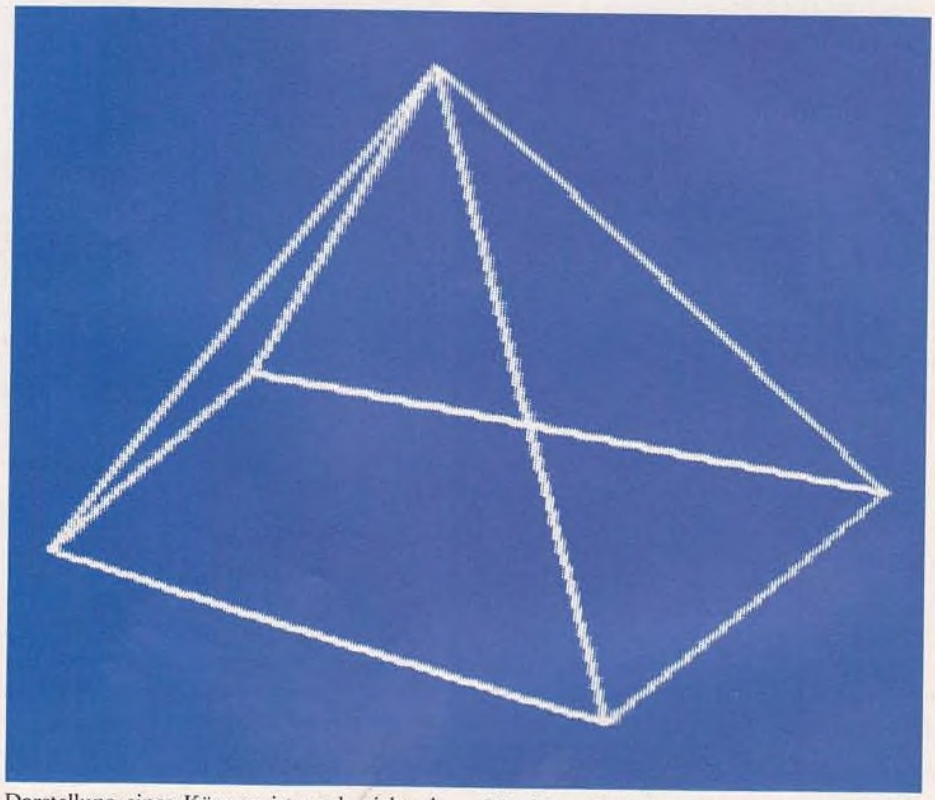
Drei Dimensionen, das sind Länge, Breite, Höhe. Das ist Raum. Und der TI 99/4A macht sich frech mit seinem 16-Bit-Prozessor in diesem Raum breit. Nach Apesoft und Kull zeigen nun auch die Männer von Hega-Soft, welch' Format der TI 99/4A hat. Schließlich sind 16 Bit nur einfach so für Spiele viel zu schade.

3-D-World

3-D-World, so heißt das Graphikpaket, das uns von den Bürostühlen kippte. Für die selbstverständlich vorhandene hohe Auflösung und Arbeitsgeschwindigkeit sorgt der TMS 9900, der vom Programm 3-D-World mit Maschinensprache so richtig in Fahrt gebracht wird.

Der TI 99/4A hat den Sprung in die dritte Dimension mit Erfolg vollzogen. Doch nun einiges zum Programm.

Das Programm 3-D-World erlaubt die Eingabe eines räumlichen Objekts und stellt es in der größtmöglichen Auflösung auf dem Bildschirm dar. Da das Programm in TMS 9900 Assembler geschrieben ist, fallen die notwendigen mathematischen Operationen zeitmäßig nicht ins Gewicht, vorausgesetzt, der dargestellte Körper weist keine allzu komplizierten Strukturen auf. Die räumliche



Die dritte Bildschirm-Dimension wird mit der 3-D-World auf dem TI 99/4A Wirklichkeit: Aus dem Dreieck entsteht eine sich drehende Pyramide.

Darstellung eines Körpers ist noch nicht alles, was dieses Programm kann. Doch dazu später. 3-D-World kann unter Extended BASIC, Editor/Assembler oder mit dem Mini-Memory geladen werden.

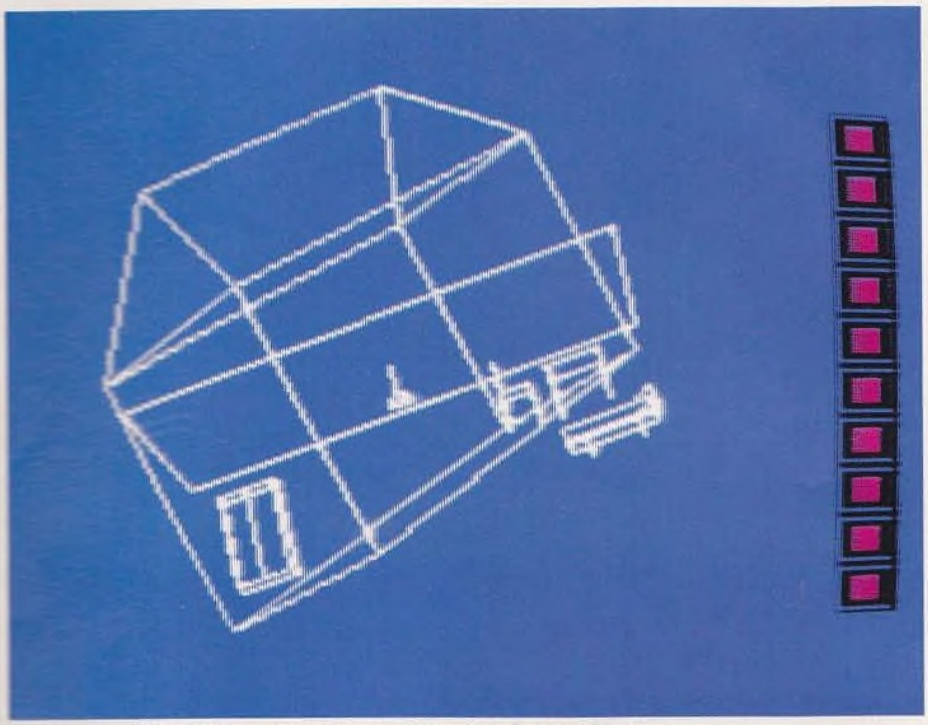
Ist das Programm einmal drin, zeigt sich als erstes ein langes Menü. Insgesamt neun verschiedene Optionen stehen dem User zur Verfügung:

▷Load Data
▷Save Data
▷Save Image
▷Show Image
▷Edit Points
▷Clear Points
▷Edit Structures
▷Clear Structures
▷Edit Colors

Zum Lieferumfang gehört auch ein ausführliches Manual, in dem von zwei Demos die Rede ist. Wählen wir also vom Menü die Nummer 1. Ruckzuck ist Demo 1 geladen und wir wählen die Nummer 4 vom Menü.

Was wir da zu sehen bekommen, enttäuscht uns zuerst: Ein simples Dreieck auf schwarzem Grund. Durch Drücken der Taste 1 beginnt das Dreieck, sich zu drehen, und zwar um die Achse, die senkrecht auf dem Bildschirm steht. Taste 2 dreht den Körper um die Hochachse. Welch ein Anblick! Das ach so simple Dreieck entpuppt sich als eine Pyramide, die auch noch um ihre Querachse zu drehen ist! Ein perfekter räumlicher Eindruck. Der Körper läßt sich tatsächlich um alle Raumachsen drehen und begutachten. Außerdem kann man, wenn man will, die Lage des Körpers im Raum verschieben. Ein Raum hat ja drei Achsen: x,y und z, und entlang dieser Achsen ist der Körper völlig frei verschiebbar.

Und wenn Ihnen die Proportionen des



Aufbau und Einrichtung eines Hauses sind einfach darstellbar

Körpers nicht gefallen, dann haben Sie die Möglichkeit, durch entsprechenden Tastendruck die x-, y- oder auch z-Ausdehnung zu verändern. Das führt manchmal zu grotesken Formen.

Ebenso kann der Körper gezoomt werden, d.h. vergrößert oder verkleinert, wobei die augenblicklich vorhandenen Proportionen erhalten bleiben. Wenn dann alle Manipulationen beendet sind, zeigen die möglichen Körperdrehungen, was Sie angestellt haben. Sind die Schwenks zu langsam, kann die Drehgeschwindigkeit des Körpers sofort erhöht werden. Von einer fließenden Bewegung kann dann allerdings auch nicht die Rede sein. Die schnellen, ruckartigen Rotationen erschweren auch das räumliche Mitdenken des Users, machen es bei komplizierteren Körpern unmöglich. Darum sollte man die Struktur des Körpers die Rotationsgeschwindigkeit bestimmen lassen.

Das Programm 3-D-World bietet die Möglichkeit, 10 Strukturen unabhängig voneinander zu entwerfen und zugleich darzustellen. Die Demo 2 zeigt ein solches Beispiel in Form eines Hauses mit einigen Einrichtungsgegenständen. Das komplette Haus besteht aus mehreren Strukturen, die einzeln optisch aus- bzw. eingeschaltet werden können. Als Kontrolle dient eine eingeblendete Strukturleiste, die aus 10 Kästchen besteht. Der obere Kasten entspricht der Struktur 1 und der letzte der Struktur 10 bzw. 0. Um die jeweilige Struktur auszuschalten, muß nur die entsprechende Ziffer gedrückt werden. Dann verschwindet die Struktur vom Schirm, und das entsprechende Kästchen der Strukturleiste wird schwarz. Erneutes Drücken der Ziffer schaltet die Struktur wieder ein.

Definition des Raums

Auf diese Art und Weise können komplizierte räumliche Zusammenhänge sehr einfach und anschaulich dargestellt werden. Natürlich kann der gesamte Körper weiterhin gedreht, verschoben, gedehnt, gestaucht, verkleinert und vergrößert werden. Ausgeblendete Strukturen kann man wahlweise beim Schwenken unsichtbar mitdrehen oder an ihrer alten Position stehenlassen. All dies führen uns die beiden Demos vor.

Wer glaubt, nun sofort an die Arbeit gehen und phantastische Darstellungen selbst auf den Bildschirm bringen zu können, den müssen wir leider enttäuschen. Ohne gründliche Vorarbeit läuft gar nichts! Das Programm laden und drauflosentwerfen ist nicht. Zuerst sollte der Bleistift gespitzt werden.

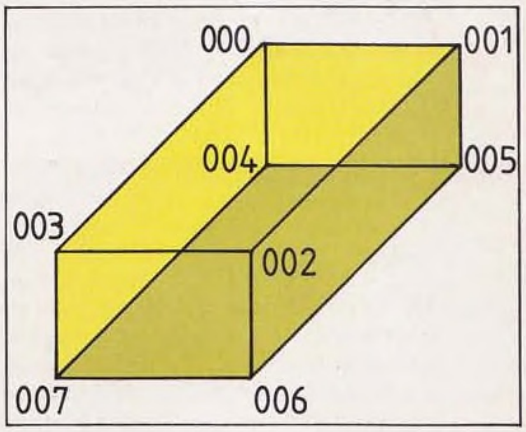
Jeder Körper besteht aus Punkten und jeder Körper hat drei Hauptachsen. Somit hat jeder Punkt eines Körpers eine eindeutige Lage zu diesen Achsen. Und diese Lage im Raum muß von Ihnen definiert werden.

Also zuerst ein Koordinatenkreuz aufzeichnen, mit x- und y-Achse. Im Winkel von 45° die z-Achse zeichnen und jetzt die Punkte Ihres gedachten Körpers einzeichnen. Nun die Abstände zur x,y- und z-Achse ermitteln und aufschreiben, denn das menschliche Gedächtnis ist schwach und leider auf Dauer auch unzuverlässig.

Dann die Nummer 5 des Programm-Menüs anwählen (Edit Points). Jetzt sieht man vier Spalten: N,x,y,z. N ist dabei die Nummernspalte. Jeder Punkt erhält eine dreistellige Zahl, z.B. 001. Nun müssen die Wertepaare für x,y und z eingegeben werden. Eventuell mit negativen Vorzeichen. Sind endlich alle Punkte eingegeben, ist die Arbeit noch lange nicht zu Ende. Woher soll denn der Rechner wissen, welche Punkte verbunden werden sollen?

Also weiter mit Fctn. Back ins Menü und die 7 (Edit Structures) angewählt. Welche der 10 möglichen Strukturen wird gefragt? Die erste natürlich. Also wird die 1 gerückt. Oh, Schock: Eine Maske aus zehn Zeilen und sechs Spalten erscheint. Keine Bange, das ist normal. Hier müssen die Nummern der Punkte in der Reihenfolge eingegeben werden, wie sie verbunden werden sollen.

Punkte dimensionieren den Quader



Ein Beispiel:

Um aus den obigen Punkten einen Quader werden zu lassen, geben wir ein:
000 001 002 003 000 004 005 006 007
004 MOV 001 005 MOV 002 006 MOV
003 007

Die Eingabe von MOV bewirkt, daß z.B. von Punkt 004 zu Punkt 001 keine Linie gezogen wird. Der Rechner merkt sich den Punkt 001 als Anfangspunkt einer neuen Linie! Durch zweimaliges Drücken von Fctn. Back kommt man zurück ins Menü, wo Sie

Und ewig lockt der Prozessor...

jetzt die 4 (show imagine) anwählen sollten. Nun können Sie Ihre Schöpfung von allen Seiten begutachten und gegebenenfalls nach Belieben verändern.

Für das einfache Quaderbeispiel sind nur acht Punkte erforderlich. Wieviele Punkte müssen aber festgelegt werden, wenn Figuren wie Kreise, Ellipsen oder Kurvenstücke definert werden sollen? Hier zeigt sich die Schwäche des Programms. Vordefinierte Zeichen sind nicht im Programm enthalten, so daß Kreise oder Kurven sehr lange Entwicklungsphasen benötigen und Sie einige Stunden konzentrierter Arbeit kosten.

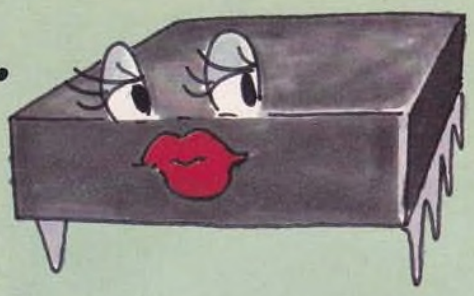
Ebensowenig können die Flächen zwischen den Linien ausgefüllt werden. Darunter leidet die Anschaulichkeit, denn gezielt ausgefüllte Flächen unterstützen das mangelhafte räumliche Vorstellungsvermögen, unter dem viele Zeitgenossen leiden. Die Farbwahl beschränkt sich auf den Vordergrund, Hintergrund und die Achsenfarbe. Bei einer Abspeicherung der Figur auf Diskette bleibt die Farbe gänzlich unberücksichtigt, sie geht einfach verloren.

Etwas luschig: das Manual

Alle Funktionen werden beschrieben und ausreichend erklärt. Leider fehlt es am festen Einbanddeckel und das Manual sieht schnell schmuddelig aus. Die einfache Heftklammer ist mit Sicherheit nicht der Weisheit letzter Schluß. Die einzelnen Blätter lösen sich schnell und sind dann unweigerlich verloren.

Der Satz "Ohne Fleiß kein Preis" gilt bei 3-D-World ganz besonders. Aber die schwere Vorarbeit wird durch das Ergebnis mehr als wettgemacht. Daß Ihr Gebilde auch gespeichert werden kann, ist selbstverständlich. Sie haben die Wahl zwischen Diskette oder Cassette. Leider fehlt die Möglichkeit, die im Bit Map Mod erstellten Bilder als Hardcopy auszudrucken. Dem fortgeschrittenen User dürfte aber die Erstellung einer entsprechenden Druckerroutine nicht allzu schwer fallen.

Alles in allem handelt es sich bei 3-D-World um ein hervorragendes Diskettenprogramm, das wir all denen empfehlen möchten, die mehr aus ihrem TI herausholen wollen und auch bereit sind, einiges an Kopf-Vorarbeit zu leisten. Denn das berühmte "Mach mal eben" ist bei 3-D-World wirklich fehl am Platze. Aber wer exakt arbeitet, dem erschließt sich die phantastische Welt der dritten Dimension, und das zu einem Preis von rund hundert Mark. Irgendwelche Kommentare? Gute Software muß nicht unbedingt teuer sein!



Graphic Master

Das zweite Graphikpaket, das wir für Sie getestet haben, heißt Graphic Master und stammt ebenfalls von Hega-Soft. Mit diesem auf Diskette gelieferten Programm kann man mit Hilfe selbstdefinierter Zeichen oder im direkten Zeichenmodus Farbgraphiken erstellen. Graphic Master hat eine hohe Auflösung von 232 x 168 Punkten, die uneingeschränkt angesprochen werden können. Jeweils acht zusammenhängende Punkte können einer Vorder- oder Hintergrundfarbe zugewiesen werden. Denken Sie daran: der TI hat 16 Farben zur Auswahl. Graphic Master ist wie 3-D-World ein reines Maschinenprogramm und benötigt deshalb natürlich die 32 KRAM. Es kann unter Extended BASIC, Editor Assembler oder Mini-Memory geladen werden.

Hat Ihr Rechner das Programm intus, erscheinen das Zeichenfeld und eine Menüleiste am rechten Bildschirmrand. Diese Menüleiste besteht aus sieben Bildfeldern, die einzelne Programmfunktionen darstellen sollen. Sie haben nun die Möglichkeit, über einen Joystick oder die Tastatur ein blinkendes Feld in der Menüleiste hin und her zu schieben. Das obere Menüfeld trägt einen Buchstaben und dient als Umschalter in ein anderes Menü. Insgesamt gibt es davon fünf! Haben Sie das blinkende Feld auf den Umschalter geschoben und den Feuerknopf gedrückt, erscheint das nächste Menü. So einfach geht das auch bei allen anderen Programmfunktionen, die das Menü beinhaltet.

Von besonderem Interesse ist natürlich das Draw Menü. Die Funktionen, die hier angeboten werden, erlauben das direkte Zeichnen auf dem Bildschirm. Ein kleiner, weißer Rahmen fordert dazu auf, einen Punkt für eine Linie zu setzen. Nachdem dies geschehen ist, kann über Joystick oder Tastatur der Rahmen über den Bildschirm gescheucht werden. Vorsicht! Zu Anfang bewegt sich der Rahmen noch langsam, startet aber dann durch, und wenn man nicht aufpaßt, rast er vom Schirm. Der zweite Punkt der Linie ist gesetzt, und ruckzuck ist aus den beiden Punkten eine schöne Linie geworden. Wer jetzt geschickt ist, kann wunderschöne Kunstwerke erschaffen, die ihn allerdings auch wunderschön viel Schweiß kosten. Über das Color Menü kann das Kunstwerk colo-riert und mitsamt Farbe auf Diskette für spätere Verwendung gespeichert werden. Wem die Farben nicht gefallen, kann sie in den Mülleimer werfen, dann steht das Kunstwerk wieder nackt da. Oder Sie können das ganze Bild löschen, was aber nicht immer ratsam erscheint, da man die Möglichkeit hat, einzelne Bildpartien gezielt zu löschen.

Besonders schön ist der Umstand, daß über ein Shape Menü eigene Zeichen (16 x16 Punkte) selbst definiert werden können. Bis zu 100 Symbole können sich auf einmal im Rechner befinden, gezielt ausgewählt und an irgendwelchen Bildschirmstellen placiert werden. Dank der Joystick- und Menüsteuerung ist dieser Vorgang recht einfach zu handhaben. Leider besitzt Graphic Master keine Druckerroutinen. Ihre Kunstwerke können Sie also nicht als Postkartengrüße besonderer Art um die Welt schicken. Hier ist Do It Yourself angesagt.

Aufgepeppte Adventures

Bevor man aber anfängt, wild drauflos zu zeichnen, sollte man das 16 Seiten starke Manual lesen. Hier werden alle Menüs mit ihren Funktionen erklärt und beschrieben. Wer alles verstanden hat und ein wenig herumprobiert, der wird schnell die gewünschten Erfolge erzielen — Farbfernseher vorausgesetzt. Das Manual geht sogar so weit, zu erklären, wie das fertige Bild abgespeichert werden kann. Damit ist es auch, genau wie 3-D-World, möglich, die so entstandenen Werke von anderen BASIC-Programmen aus zu laden und einzubinden. Beispielsweise für die Illustration von Adventures oder anderen Spielen.

Damit ist Graphic Master ein Werkzeug, das Ihnen enorme Möglichkeiten bietet, für einen vergleichsweise geringen Preis von einem knappen Hunderter.

Und wer jetzt noch meint, es gäbe keine gute Software für den TI 99/4A, der gehört mit Sicherheit nicht zur Leserschaft von COMPUTER Praxis/Telematch.

Die Programme 3-D-World und Graphic Master gehören zu der angeblich nicht vorhandenen guten Software.

Jeder User, der Wert auf Graphik legt, sollte je ein Exemplar dieser beiden Pakete in seine Programmsammlung aufnehmen. Man kauft sich damit zwei Programme, an denen man mit Sicherheit sehr lange viel Spaß haben dürfte.

K. W.

In letzter Minute!
Wie uns die Firma Computer-Hüsli mitteilt, gibt es jetzt für die Programme Graphic-Master, 3-D-World, Ex-BASIC II und den Painter eine Hardcopyroutine. Die Preise bewegen sich zwischen 20 und 30 Mark.

SOFTWARE, DIE SPASS MACHT

APPLE //, //e, //+

Quest for tires	Disk	129,–
Pole position	Disk	129,–
Summer games	Disk	129,–
Zaxxon	Disk	129,–
Print shop	Disk	139,–
Cut and paste	Disk	159,–

ATARI COMPUTER

Archon I	Disk	99,–
Archon II	Disk	99,–
Mask of the sun	Disk	109,–
Pitfall II	Cas.	49,–
Lode Runner	Disk.	129,–
Defender	Mod.	129,–
Dig dug	Mod.	129,–
Zaxxon	Mod.	129,–
Bruce Lee	Disk + Cas.	139,–
Bank street writer (dt.)	Disk	249,–
Cut and paste	Disk	159,–

COMMODORE 64

Archon I	Disk	99,–
Archon II	Disk	99,–
Beyond castle Wolfenstein	Disk	129,–
Quest for tires	Mod.	139,–
	Disk	129,–
Decathlon	Cas.	49,–
	Disk	79,–
H.E.R.O	Cas.	
Seven Cities of Gold	Disk	99,–
Bank street writer	Disk	249,–
Structured basic	Mod.	189,–
Cut and paste	Disk	159,–
Magic Desk	Mod.	239,–

COMMODORE VC 20

Lode Runner	Mod.	69,–
Coplifter	Mod.	69,–
A.E.	Mod.	69,–
Gridrunner	Mod.	69,–

SPECTRUM

Beach Head	Cas.	49,–
Jet Set Willy	Cas.	24,–
Enduro	Cas.	49,–
H.E.R.O.	Cas.	49,–
River Raid	Cas.	49,–

Versandkosten bei Vorkasse oder Bestellwert über DM 160,–. Bei Nachnahme + 4.90.

Katalog mit Spiel- und Arbeitssoftware gleich anfordern:

TELEDIENST
Mainzer-Tor-Anlage 45
6360 Friedberg
Telefon 06031/91650
Btx. 213213

POST

Die liebe Kompatibilität

Ich möchte mir einen C16 von Commodore kaufen. Laufen Eure C64-Listings auch auf diesem Gerät?
Andreas Schneider, Frankenberg

Der C16 ist mit dem C64 nur teilweise kompatibel. Die Bildschirmaufteilung, die Steuerzeichen usw., sind beim C16 gleich geblieben. Allerdings verfügt er über einen anderen Prozessor. Es können also Probleme bei Peeks und Pokes auftreten. Wer sich aber auf dem 6510 auskennt und sich in den 7501 einfuchst, wird klarkommen. Allerdings kann der C16 keine Sprites darstellen und hat einen verringerten Tonumfang.
Wir werden Sie über Rechnerneuerscheinungen auf dem Laufenden halten und Ihnen Tips und Tricks für alle Commodores im Heft liefern.
Wer Schwierigkeiten beim 'Übersetzen' eines Programms hat, wende sich vertrauensvoll an uns — mit Telefonnummer. Wir helfen ganz konkret weiter. *CP/T*

Der Nepp geht um?

Vor einiger Zeit bestellte ich bei der Firma XY eine Speichererweiterung. Als ich das Modul bekam, stellte ich fest, daß das Modul unbrauchbar war: Statt der notwendigen fünf Schalter hatte es nur vier!
Ich schickte die Erweiterung zurück mit der Forderung, mir mein Geld zurückzusenden. Nichts geschah. Ich rief an, und mir wurde gesagt, daß sowas 2—3 Wochen dauert, dann legte man sehr eilig auf.
Nach einem Monat hatte sich XY immer noch nicht gemeldet. Ich schrieb also erneut. Keine Reaktion. Nach Monatsfrist telefonierte ich wieder mit XY.
Bei diesem Anruf wurde mir unverschämt nahegelegt, noch mal zu schreiben. Darauf ging ich nicht ein und gab dem Versand XY erneut die Frist von einer Woche. Noch immer hat sich nichts getan.
Carsten Drewes, Leopoldshöhe

In solchen Fällen können wir raten, die Verbraucherzentralen, die es in vielen deutschen Städten gibt, um Hilfe oder Auskunft für den konkreten Fall zu ersuchen. Sonst läßt man aus Unwissenheit vielleicht wichtige Fristen verstreichen. In diesem konkreten Fall, in dem uns nur die Stellungnahme einer Seite vorliegt, möchten wir nicht urteilen und nennen auch deshalb den Namen des beschuldigten Unternehmens an dieser Stelle nicht.
Wer aber ähnlich unangenehme Erfahrungen sammeln mußte, den bitten wir, sie uns mitzuteilen. Sollte sich dadurch ein Unternehmen herausstellen, bei dem derart miese Taktiken zur Tagesordnung zu gehören scheinen, werden wir im Interesse unserer Leser nachrecherchieren und in unserem Magazin ein offenes Wort sprechen. *CP/T*

Teures Vergnügen?

Ihr habt "Mapping the Atari" als Spitzen-Buch vorgestellt. Ich habe versucht, es in Essen zu bekommen. Für etwa hundert Mark hätte man es mir bestellen können. Ihr nanntet als Preis 58 Mark. Wie geht das an?
Noch eins. Ich habe CP/T abonniert. Heft 8/9 von Computer Praxis war eine Doppelnummer. Bekomme ich eine Zeitschrift weniger, als ich bezahlt habe?
Andreas Busch, Essen

Keine Sorge, liebe Abonnenten. Zwölf Hefte sind und bleiben zwölf Hefte. Ihr werdet um keines betrogen.
Den Atari-Klassiker bekamen wir von einem Hamburger Buchladen, der wohl recht gute 'Connections' in die Staaten hat — darum billig. Für Sie wichtig zu wissen, daß Boysen und Maasch, Hermannstr. 31, 2000 Hamburg 1, 040-30050515 auch einen Postvertrieb unterhält. *CP/T*

aus: Computer-Sti(h)cheleien

Wo gibt's die Software

Ihr Artikel zum 'Flugsimulator 2' hat es mir angetan. Leider erwähnten Sie nicht, was das Spiel kostet und wo man es bekommt.
Rainer Lozynski, Karlsruhe

Software und Geräte, die wir testen, sind oft beim Händler um die Ecke erhältlich. Da es keine Preisbindung gibt, können wir meist (große Ausnahme: Bücher) über Preise nichts genaues sagen. Verraten können wir allerdings, wie man an genaue Informationen kommt. Für den Flugsimulator wende man sich an Computer Plus Soft, Bahnstr. 22-26, 4220 Dinslaken, 02134-7905.
Seit Heft 10/84 nennen wir übrigens die wichtigen Adressen in unserem Bezugsquellennachweis vor der letzten Umschlagseite. *CP/T*

Selbstgebaute Programme

Seid Ihr an selbstgeschriebenen Programmen interessiert? Wenn ja, schicke ich sie Euch gerne. Gleich mit diesem Brief sende ich sie Euch jedoch infolge schlechter Erfahrungen nicht, da es bei vielen Eurer Konkurrenten bei Desinteresse nicht nur "GOTO 999", sondern auch "IF TAPE THEN GOTO private Cassettensammlung : IF MANUSKRIPT THEN GOTO Papierkorb : IF Copyright THEN NOT GOTO back to the author

Das ist Ihre Seite, liebe Leser. Forum der Diskussionen. Platz für Fragen und Antworten, Podium für Kritik, Zustimmung, Vorschläge, Anmerkungen und Anregungen. Und sicher auch Ausgangspunkt für den Dialog zwischen Computerspielfreunden. Brieflich oder gar persönlich. Auch bei CP/T gilt: Je kürzer die Zuschrift, desto einfacher haben wir's mit der Veröffentlichung. Und wie üblich müssen wir uns vorbehalten, Briefe gekürzt oder auszugsweise wiederzugeben. Also: Ab geht die Post

an CP/T
Redaktion
Paulstraße 3
2000 Hamburg 1

: IF Honorar THEN GOTO eigene Tasche“ heißt...
Namen nenne ich lieber nicht, die Fälle sind aber tatsächlich vorgekommen.
Wolf-Dieter Roth, Gilching

Solche, wollen wir doch mal Deutsch reden, Schweinereien gibt es bei uns gottlob nicht. Wir sind immer an tollen Leserlistings interessiert, schon im Interesse unserer Leser. Zur Einsendung genügt es uns, das vorgeschlagene Programm auf Disk oder Cassette zu erhalten. Dazu eine kurze Beschreibung und — wichtig! — die Telefonnummer des Einsenders, damit wir bequem mit ihm Kontakt halten können.
Natürlich können wir auch nicht mehr Listings veröffentlichen, als wir Platz im Heft haben. Etwas Geduld bitten wir darum auch mitzubringen. Verschollen geht aber bei uns nichts. CP/T

TI-Invers

Ich möchte Sie zu Ihrer Computerzeitschrift beglückwünschen. Als Dankeschön hier ein Tip für TI 99/4A-Besitzer mit Ex-Basic:
CALL CLEAR :: CALL SCREEN (2):: FOR I=0 TO 14 :: CALL COLOR (I,16,2):: NEXT I :: RUN ENTER
So erzielt man inverse Schrift, solange kein Programm im Speicher ist, obwohl der TI warnt: “*Warning no program present“. Auch die Spritebehandlung ist mit diesem Trick außerhalb eines Programms möglich:
CALL CLEAR :: CALL MAGNIFY(2) :: CALL SPRITE (#1,42,2,100,100,20,0):: RUN ENTER
Helge Schlick, Dornstadt

Über derart kommentierte Dankeschöns freuen wir uns natürlich besonders. CP/T

Die neuen Activision-Cassetten für Sinclair und MSX-Computerfreunde!

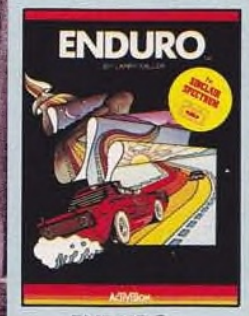
RIVER RAID – JAGDFLIEGER

Lieferbar für SINCLAIR und MSX

H.E.R.O. – HELIKOPTERHELD

PITFALL II – DIE SUCHE NACH DEM UNTERIRDISCHEN SCHATZ

GHOSTBUSTERS – DIE GEISTERJÄGER



ENDURO – TRANSAMERIKA-RENNEN

THE DESIGNER'S PENCIL

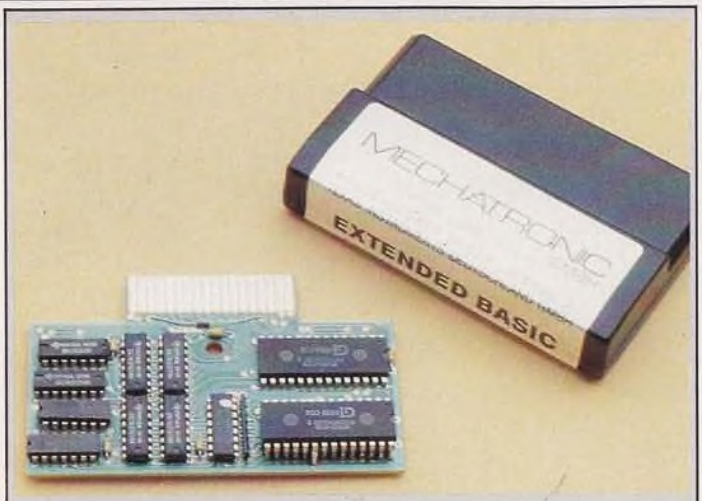
Lieferbar für SINCLAIR

SPACE SHUTTLE – EINE REISE INS ALL

ZENJI

ariolasoft
Qualität ist unser Programm!

ACTIVISION
HOMECOMPUTER-SOFTWARE



"Neues" Extended Basic für den TI!

Unter der Lizenz von Texas Instruments Deutschland GmbH fertigt die Sindelfinger Fa. Mechatronik wieder das Extended Basic. Damit ist eines der wichtigsten Module für den TI 99/4A auch nach dem Rückzug von TI weiterhin erhältlich. Dieses Modul unterscheidet sich von dem früheren TI Extended Basic nur beim Auswählen im Titelbild. "Mechatronic/PS Extended Basic" ist dort jetzt zu finden. Der interne Aufbau des Mechatronic Extended Basic ist vom Original verschieden, aber das ist für den Benutzer sicherlich unerheblich, denn die Funktionsweise entspricht exakt der des TI Extended Basic. So laufen alle Extended Basic Programme auch auf diesem Modul. Befürchtungen, das neue Extended Basic könnte langsamer arbeiten, sind völlig unbegründet, die Ausführungsgeschwindigkeit ist identisch. Geliefert wird das Extended Basic von Mechatronic zusammen mit dem bekannt guten Programmierhandbuch "Extended Basic für Anfänger und Fortgeschritene". Der empfohlene Verkaufspreis liegt mit 259.90 DM erfreulicherweise unter dem noch vor Jahresfrist für das Extended Basic verlangten Preis.
Bezugsquelle: Fachhandel oder Mechatronic, Dresdner Str. 21, 7032 Sindelfingen.

Dem Sinclair-QL nun auch noch seine Floppy

Unschön war von Anfang an, daß dem QL das Microdrive zugewiesen wurde; die Kapazitäten stehen in schlechtem Verhältnis. Quest Automation bietet in England jetzt das an, was allzu lange fehlte. Die Shugart-kompatiblen Floppy-Laufwerke werden als Single ab 200 KByte zu etwa tausend Mark, dual bis 800 KByte zu umgerechnet 2400 Mark angeboten. In England gibt's nun auch das Digital Research CP/M-68K für den QL; als Floppy (ca. 100 Mark), als Microdrive-Cassette (ca. 320 Mark).
Bleibt zu hoffen, daß sich auch auf dem Festland bald ein Vertriebspartner für diese interessanten Neuheiten findet. Im Sinne der User.

TELESPIELE·SPASS ZU KNÜLLER·PREISEN!

TELESPIELE VON IMAGIC
von
Electronic Vertrieb GmbH
Friedrich-List-Straße 1
7012 Fellbach

TELESPIELE VON IMAGIC

Atari® 2600

Best.-Nr.	Titel	Preis DM
01	Demon Attack	
02	Star Voyager	
03	Atlantis	
04	Cosmic Arc	
05	Riddle of the Sphinx	
06	Dragonfire	NEU
		nur je 29,95
07	No Escape	NEU
08	Fathom	NEU
09	Quick Step	NEU
10	Moonsweeper	NEU
		nur je 49,–

MATTEL®-Intellivision

Best.-Nr.	Titel	Preis DM
11	Beauty and the Beast	
12	Micro Surgeon	
13	Swords and Serpents	
14	Dragonfire	NEU
15	Ice Treck	NEU
16	Dracula	NEU
17	Nova Blast	NEU
18	Safe Cracker	NEU
19	White Water	NEU
		jedes Spiel 35,–
		drei Spiele nur 99,–

Coleco®

Best.-Nr.	Titel	Preis DM
20	Moonsweeper	NEU
21	Nova Blast	NEU
22	Wing War	NEU
		jedes Spiel 89,–

Philips® G 7000

Best.-Nr.	Titel	Preis DM
23	Demon Attack	
24	Atlantis	
		jedes Spiel 39,–

TELESPIELE-SPASS-BESTELLCOUPON

Hiermit bestelle ich

☐ per Nachnahme (+4,50 DM Postgebühr)
☐ per Euroscheck (liegt bei)

● Ausland nur gegen Vorkasse

Paket-Nr.	Anzahl	Preis
		DM
		DM

Bestell-Nr.	Anzahl	Preis
		DM
		DM
		DM

Name _____ Vorname _____
Straße _____
PLZ/Ort _____ Telefon _____

Einsenden an:
Electronic Vertrieb GmbH
Friedrich-List-Straße 1
7012 Fellbach

Über200 Adventure-Games im weitesten Sinne sind derzeit weltweit im Angebot. Spiele oder Simulationen, die sich vom üblichen Bildschirm-Geballer dadurch unterscheiden, daß vornehmlich der Kopf beim Spiel gefordert wird. Mehr noch: Adventures jedwedes Genres sind Programme, die vorzugsweise im Team gespielt werden, also die Kommunikation fordern.

Auf Fantasy- und Rollenspiele sind wir im Januar bis April dieses Jahres ausführlich in Telematch mit einem Special eingegangen, haben Spielgattungen vorgestellt und die Verbindungen zu eher traditionellen, aber reizvollen Spiel auf dem Brett mit Zinn-Miniaturen aufgezeigt. Nachstehend geben wir noch einmal für Einsteiger eine Übersicht der nach unserer Auffassung interessanten, vor allem auch der neuen Adventures, die Langzeit-Spielspaß garantieren.

Graphic-Adventure: Spaß für Fantasy-Freunde

Was aber heißt eigentlich Adventure? Der Spieler sieht sich mit einer komplexen Aufgabe konfrontiert, die — je nach Programmtyp — in einem bestimmten Zeitraum oder in unbegrenzter Zeit zu lösen ist. Diese Aufgabe kann strategischer Art sein, etwa bei "Guns Of Fort Defiance" oder "Andromeda Conquest" (beide von Avalon Hill); dabei hat die grafische Aufbereitung untergeordneten Stellenwert, begnügt sich mit abstrakten Symbolen (etwa taktischen Zeichen im übertragenen Sinne). Sie kann bei einer weiteren Spielart des strategischen Abenteuers sogar ganz auf Grafik verzichten (so z.B. bei dem deutschen Adventure-Oldie "Das Schloß des Grauens") und läuft somit als reines Text-Adventure. Weitgehend bevorzugt wird indes heute jener Spieltyp, in dem Grafik (Standgrafik und animierte Grafik), Textpassagen und Aktion, abhängig vom jeweils erreichten Spielstadium wechseln. Herausragendes Beispiel dafür ist das von dem 17jährigen Karsten Köper entwickelte Adventure "Mythos" (für Atari). Daneben dominiert das sogenannte "Grafik-Adventure", wie es besonders von den Programmautoren des amerikanischen Software-Hauses "Sierra" gepflegt wird. Interessant, wenngleich nicht neu, die Adventures "Ulysses" (für Apple), "Ultima" (für Apple- und Atari-Computer) sowie "Dark Crystal" (ebenfalls für Apple- und Atari-Computer).

Das ganze Spektrum von Adventure-Games wird deutlich, stellt man einmal aktuelle Programme für die verschiedenen Systeme nebeneinander, die in den letzten Wochen auf den Markt gekommen sind bzw. bei Erscheinen dieser Ausgabe vorliegen werden. So steht das Graphic-Adventure "Golden Mask Of The Sun" (für C 64) neben dem inzwischen fast schon zum Klassiker gewordenen "Dallas Quest" (ebenfalls für C 64), und buhlen Ataris deutsche Eigenentwicklungen "Globetrotter" und "Atlantis" um die Gunst des Bildschirm-Indiana Jones. Einen kombinierten Adventure/Simulationsweg haben die deutschen Atari-Programmautoren für "Karriere" entwickelt; die Spielgrundidee ist mit "Monopoly" identisch.

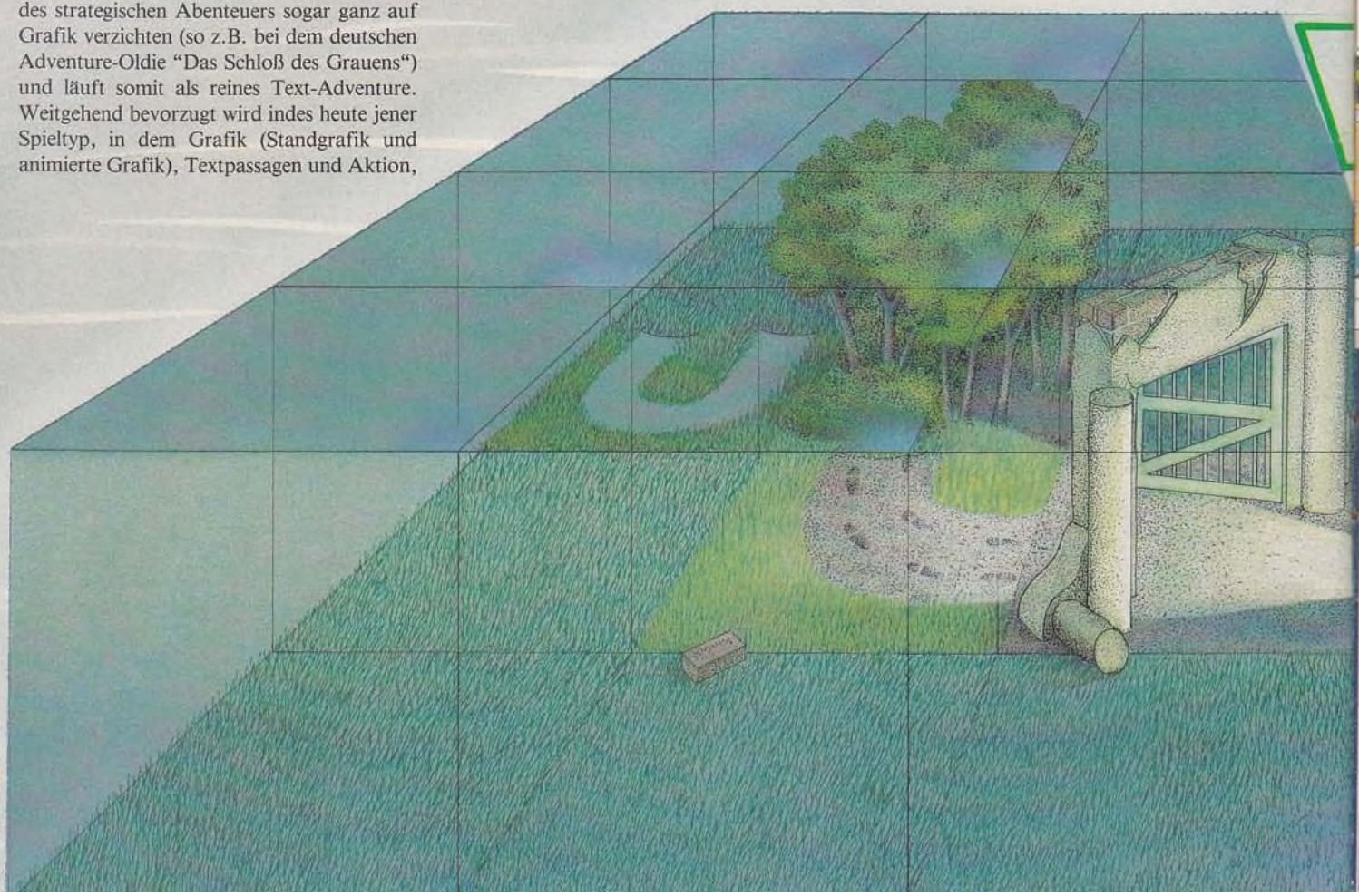
Freunde der Fantasy kommen gleichermaßen auf ihre Kosten, zumal ZX Spectrum-Besitzer: Da kann man sich z.B. ins Land der "Lords Of Midnight" versetzen lassen und es Conan nachtun (wie im Software-Review-Teil dieser Ausgabe zu lesen). "Psytron" dagegen bietet ZX-Spectrum-Freunden Gelegenheit, sich in Weltraumsimulationen zu üben — mit grafisch recht bemerkenswerten Akzenten.

Wohl mehr dem Genre Action zuzuordnen, wenngleich die Verwandschaft mit den Adventures belegbar ist, wären die zahlreichen Labyrinthspiele, die für alle Systeme in unterschiedlicher Form zu haben sind. Fast klassisches Beispiel dafür die Computerversionen der Videofolge "Pitfall" und Pitfall II". Weitaus komplexer und langfristig reizvoller sind aber Programme wie "Loadrunner" (mit beiden Fortsetzungen, dem "Championship" und "Championship II") oder der Neuling von Brøderbund — zumindest für den deutschen Markt — "The Castle Of Doctor Creep".

Abenteuer im Kopf

Gewiß: Dieser Programmtyp ist schwerpunktmäßig auf Aktion ausgerichtet. Ohne Kombinieren jedoch geht auch hier nichts. Eine Zuordnung in den Bereich scheint deshalb gerechtfertigt. Benachteiligt sind, wie so oft, Besitzer des TI 99/4A. Unser Anliegen war, ein attraktives Adventure zu schaffen, das relativ leicht einzugeben ist, darüber hinaus aber die Möglichkeit gibt, eigene Adventure-Idee zu realisieren. Zitieren wir einfach frei: "Die wahren Abenteuer sind in Ihrem Kopf". Probieren Sie's doch ganz einfach einmal aus...

h.h.



Way Home

Knisternde Spannung und wohliges Gruseln auf dem TI

Adventure heißt das Zauberwort. Für jeden Home-Computer gibt es mehr oder weniger interessante Abenteuer zu bestehen, nur bei dem TI ist das Angebot recht dürftig.

Wir wollen das ändern und stellen Ihnen hier »Way Home« vor, ein Adventure für den TI 99/4A, mit Extended BASIC und Cassetten-Recorder oder Disk.

Irgendein unliebsamer Zeitgenosse hat Sie an einen unbekannten Ort verschleppt. Es gilt, den Weg nach Hause zu finden — doch der ist mit "Abenteuern" gepflastert.

Der Witz des Programms liegt darin, daß man sich den Handlungsablauf auch selber aufbauen kann. Aus Platz-Gründen — der TI hat ja bekanntlich mit X-BASIC nur rund 13 K Speicherkapazität — wurden die Handlungs-Daten vom Programm getrennt in einer Datei auf Cassette gebracht. Damit auch Sie in den Genuß dieser Daten kommen, hier also gleich zwei Listings. Das erste ist das eigentliche Adventure, das zweite ein Programm, das eine Datei erzeugt.

Wir bieten Usern mit Disketten-Laufwerk die Möglichkeit, diese Datei auch von Diskette lesen zu können. Deshalb haben Sie in den Listings an einigen Stellen zwischen jeweils zwei Zeilen die Wahl. Die eine Zeile ist eine Datei mit »CS1«, also für Cassette, die andere mit »DSK1«, also für Diskette. Die jeweils nicht benötigte Zeile sollte unbeachtet bleiben. Die REMark-Zeilen können nur mit RAM-Erweiterung eingegeben werden.

Variablen-Listen zu Way Home

Arrays:

GEP$(5)	Array für das Gepäck
RGE$(3)	Gepäck-Erweiterung
TEXT$(8)	Situations-Texte
COL(8)	Situations-Bildschirmfarbe
L$(8,4)	Dinge, die mitgenommen werden können
SON(8,5)	Besondere Situationen
RICH$(8,4)	Die möglichen Richtungen, um eine Situation zu verlassen
BEF$(8,7)	Befehlssatz

Boolsche Variable:
RUC, ZS, WU, WD, AN, LI, ZIEL
Sie dienen der Erkennung besonderer Bedingungen und können nur die Werte '0' und '1' annehmen.

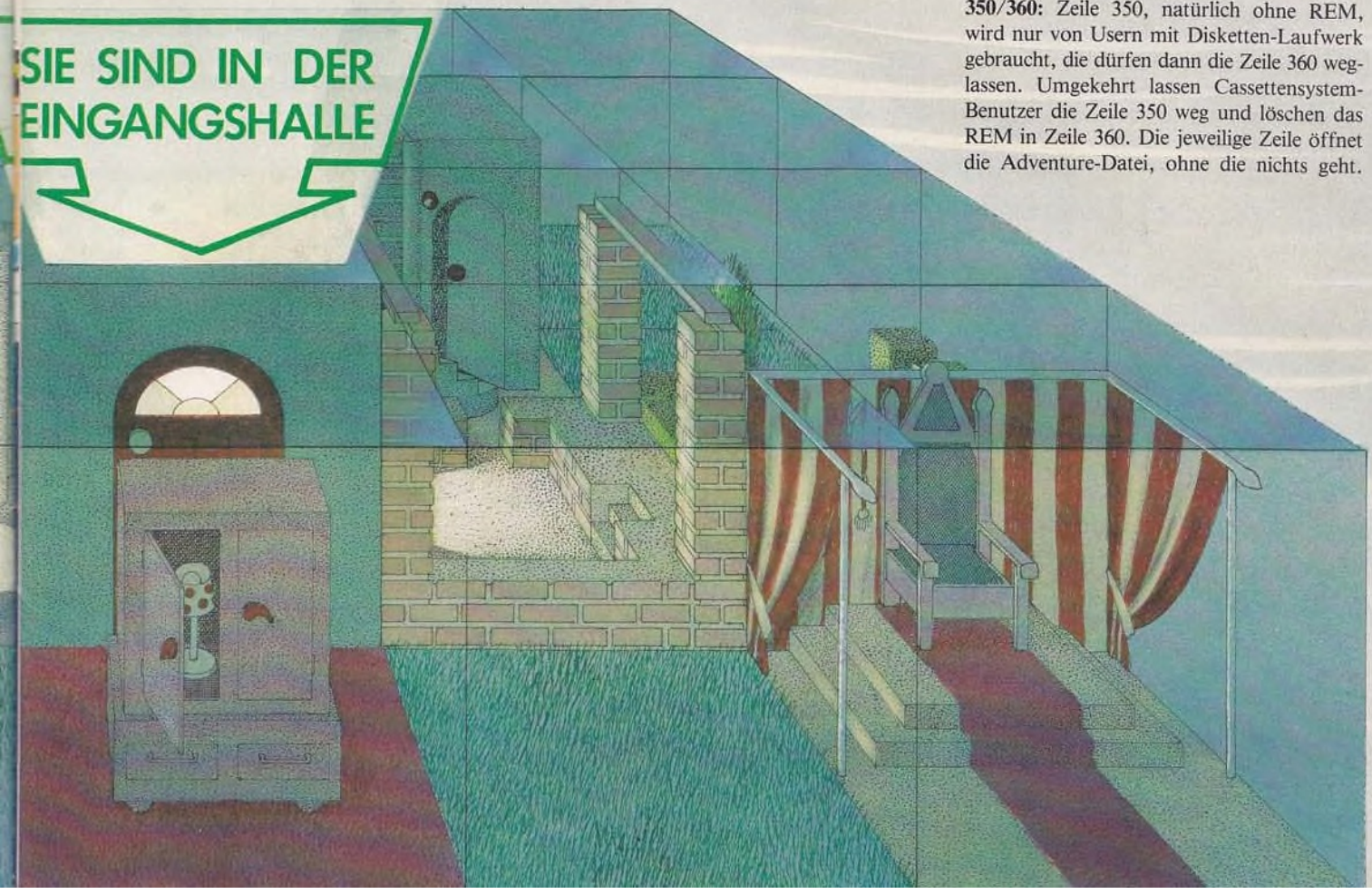
Schleifen-Variable:
I, J, BEF

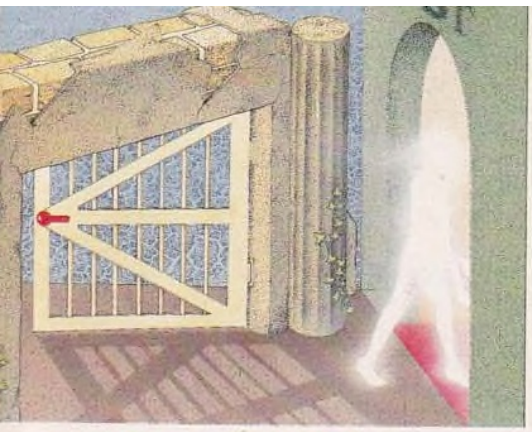
Arbeits-Variable:

PU = Punkte	BLOCK = Zähler
PUM = Highscore	ZEI = Menü-Variable
SIT = Situation	RI = Richtung
SN = neue Situation	TUN$ = Befehl

J$ = READ-Variable und Zauberwort
KEY, STATUS = Tastatur-Abfrage
ZMAX, WAS$, G$ = Gepäck-Hilfsvariable

Zum Aufbau von Way Home:
Zeile
130—200: Befehls-Kürzel. Das Programm erkennt lediglich die ersten drei Zeichen eines Befehles. Dennoch sind sie so gewählt, daß der TI fast jede Aufforderung 'versteht'.
240—280: Initialisierung. Das Programm arbeitet mit acht Arrays (Feld-Variablen). Die Bedeutung der Arrays können Sie in der Variablen-Tabelle finden. Zeile 250 liest die Befehls-Kürzel aus dem DATA-Block in das Array BEF$. Die umdefinierten Zeichen 64 und 96 haben eine wichtige Rolle im Spiel. Zeile 270 setzt alle Variablen auf den Start-Wert.
350/360: Zeile 350, natürlich ohne REM, wird nur von Usern mit Disketten-Laufwerk gebraucht, die dürfen dann die Zeile 360 weglassen. Umgekehrt lassen Cassettensystem-Benutzer die Zeile 350 weg und löschen das REM in Zeile 360. Die jeweilige Zeile öffnet die Adventure-Datei, ohne die nichts geht.





Way Home

Sie findet sich im Anhang als zweites Listing.
400—510: Weglegen. Es wird geprüft, ob überhaupt Gepäck vorhanden ist. Wenn ja, erscheint ein Pfeil, der mit den Cursor-Tasten für 'HOCH' und 'RUNTER' verschoben werden kann. Will man doch nichts weglegen, so muß man FCTN 9 (gleich 'BACK') drücken.
550—570: Prüfung, ob wohl etwas geworfen oder sogar getroffen wurde...
610—690: Es kommt vor, daß das Weglegen eines Gepäckstückes eine bestimmte Reaktion hervorruft.
730—890: Aufnehmen. Die Steuerung des Pfeils funktioniert wie in der Routine "Weglegen" (400—510). Es wird geprüft, ob noch 'Kapazitäten' frei sind, um noch mehr mitzunehmen, bzw. ob es überhaupt etwas mitzunehmen gibt.
930—1380: Situation verlassen. Auf den Befehl 'GEH' oder einer ähnlichen Anweisung und der entsprechenden Wahl aus dem Menü wird ermittelt, welche Situation folgt, ob es eventuell ungelöste Probleme gibt oder ob das Ziel erreicht wurde.
1420—1460: Wurde der richtige Weg gefunden, so wird zum einen der Punkte-Stand erhöht, zum anderen der nächste Block von Cassette bzw. Diskette eingelesen, das heißt, die nächsten acht Situationen. Wer mit einem Cassetten-Recorder arbeitet, darf diesen also nicht einfach ausschalten.
1500—1590: Das Spiel wird über eine Bildschirm-Maske mit verschiedenen Menüs gespielt. Außerdem wird geprüft, ob besondere Situationen vorliegen. Zeile 1590 nimmt die Befehle des Spielers entgegen. Möchte man die eingegebenen Befehle einfach übernehmen, muß man ganz einfach nur die ⟨ENTER⟩-Taste drücken.
1630—1690: Der eingegebene Befehl wird mit dem Array BEF$ verglichen. Kennt der Rechner den Befehl nicht, bekommt der Spieler eine neue Chance. Die Hilfsvariable BEF gibt dann an, welche Routine ausgeführt wird.
1730: Diese Zeile wird teuer, da die Bitte um Hilfe die Hälfte der Punkte kostet, die man angesammelt hat. Ist Ihnen dies zu viel, muß die Zeile entsprechend geändert werden. PU ist die Variable, die die Punkte führt.
1770—1800: Die Lampe, so man eine hat, wird an- bzw. ausgeschaltet.
1840—1930: Das Spiel wird auf den Befehl 'AUFGEBEN', 'STOP' oder 'ENDE' unterbrochen. Ist ein neuer Highscore erspielt, wird dieser auf die Cassette bzw. die Diskette geschrieben. FCTN 8, gleich REDO öffnet ein neues Spiel, CTRL C unterbricht das Programm mit STOP. Aus der Zeile 1900 oder 1910 muß man wieder, je nach System, das REM löschten.
1970—1990: Hier verbirgt sich eine kleine, aber unerfreuliche Überraschung. Die Zeile 1990 verzweigt, wenn diese Überraschung ausbleibt, nach dem gegebenen Befehl auf die entsprechende Routine.
2030—2050: Man kann auch etwas sagen — in manch häßlicher Situation hilfreich.
2090—2130: Fangen. Nicht immer ist weglaufen möglich und Messer werfen nötig.
2170—2210: Manchmal taucht ein Zwerg auf, und der benimmt sich dann auch noch total daneben.
2250—2300: Will er die Situation verlassen, wird dem Spieler ein Menü angeboten, aus dem er sich einen Weg aussuchen kann. Die Menü-Steuerung erfolgt wieder wie in der Routine 'Weglegen' (400—510).
2340—2360: Diese drei Zeilen löschen verschiedene Felder der Bildschirm-Maske.

Allgemeines zum Listing:

Aus den DISPLAY und PRINT-Befehlen sollten keine Zeichen weggelassen werden, da sie sonst eventuell nicht mehr in die Masken-Felder hineinpassen.

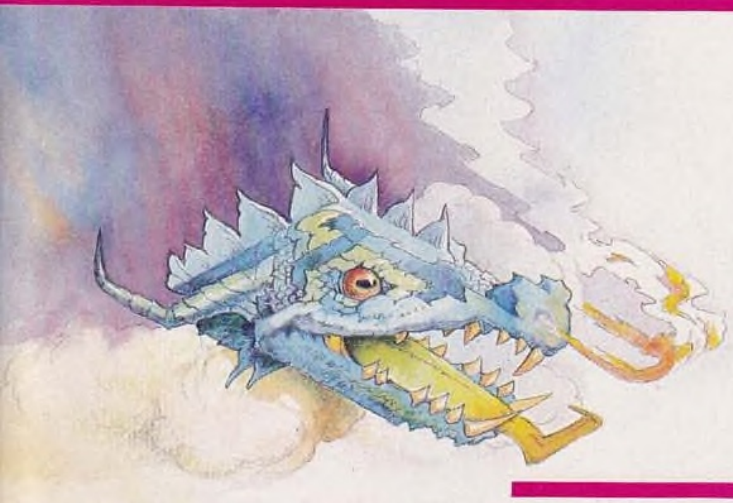
Manche Programm-Zeilen lassen sich nicht auf die übliche Weise eingeben. Mault der TI 'LINE TOO LONG', obwohl noch Zeichen folgen sollen, müssen Sie diese Tasten-Sequenz drücken:
⟨ENTER⟩, um die Zeile zu schließen. Sind noch Anführungs-Striche oder Klammern offen, müssen diese vorher provisorisch geschlossen werden.

Noch einmal ⟨ENTER⟩, um den NUMber-Modus zu verlassen.

Dann die Nummer der unfertigen Zeile eingeben und statt ⟨ENTER⟩ in diesem Fall FCTN X oder FCTN E drücken. Sonst wird sie wieder gelöscht!

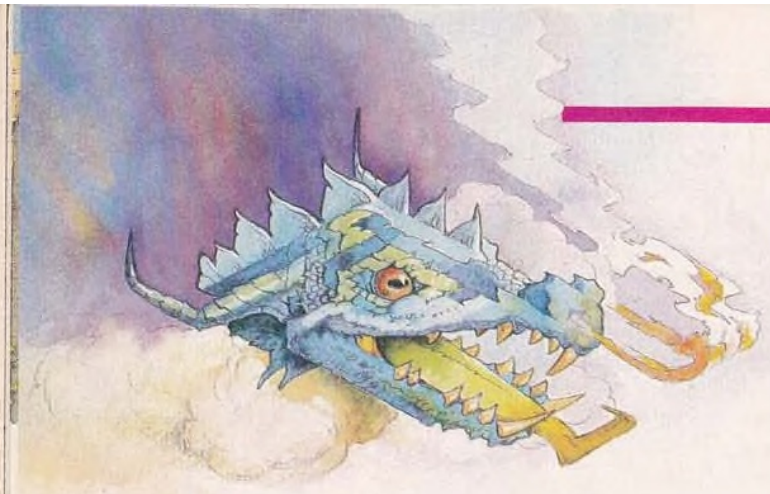
Jetzt können die restlichen Zeichen eingegeben werden. Die provisorisch gesetzten Ausführungs-Striche oder Klammern werden natürlich wieder gelöscht.
K.W.

TI-Adventure-Kid

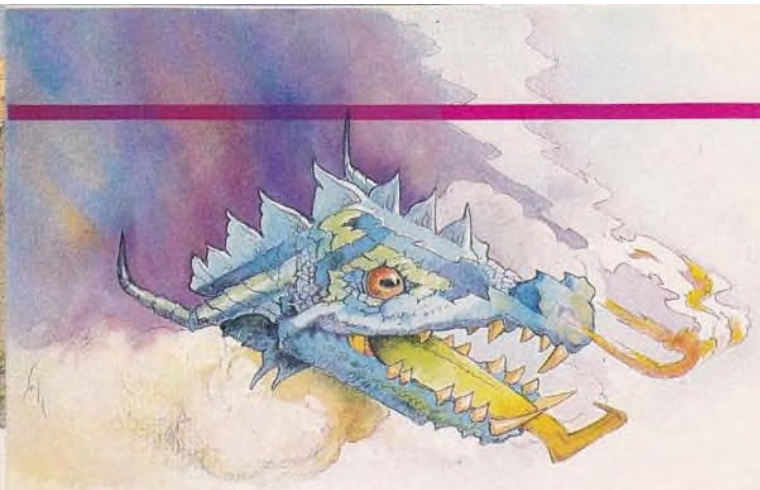


```
100 ! Alle REM's muessen ohne RAM-Pack wegfallen
110 ! VOKABEL-SATZ, ERWEITERBAR
120 !
130 DATA "HIL","HEL","","","","","",""
140 DATA "AN","LIC","AUS","","","","",""
150 DATA "STO","END","AUF","","","","",""
160 DATA "GEH","LAU","SPR","KRI","","","",""
170 DATA "SCH","WIR","LEG","STE","HAE","GIB","WER"
180 DATA "NIM","","","","","","",""
190 DATA "SAG","RUF","BRU","","","","",""
200 DATA "FAN","","","","","","",""
210 !
220 ! INITIALISIERUNG
230 !
240 RANDOMIZE :: OPTION BASE 1 ::
     DIM GEP$(5),RGE$(3),TEXT$(8),COL(8),L$(8,4),
```

```
    SON(8,5),RICH$(8,4),BEF$(8,7)
250 FOR I=1 TO 8 :: FOR J=1 TO 7 :: READ J$ ::
     BEF$(I,J)=J$ :: NEXT J :: NEXT I
260 CALL CHAR(64,"00183C7E7E3C1818")::
     CALL CHAR(96,"0018244242241818")::
     ON BREAK NEXT
270 FOR I=1 TO 5 :: GEP$(I)="" :: NEXT I ::
     FOR I=1 TO 3 :: RGE$(I)="" :: NEXT I ::
     PU,RUC,BLOCK=0 :: ZMAX=6
280 J$="" :: FOR I=1 TO 5 :: J$=J$&CHR$(25*RND+65)::
     NEXT I :: CALL CLEAR
290 !
300 ! Wahlweise Zeile 350 fuer Disketten-
310 ! oder Zeile 360 fuer Cassetten-Version
320 !
330 ! ADVENTURE-DATEI EROEFFNEN
340 !
350 REM PRINT #1:"DSK1.HIGH",INTERNAL,INPUT ::
     INPUT #1:PUM :: CLOSE #1 :: OPEN #1:
    "DSK1.ADDAT",INTERNAL,INPUT ,FIXED 192 ::
     GOTO 1420
360 REM PRINT "* PUT IN DATA-CASSETTE   CS1" ::
     OPEN #1:"CS1",INTERNAL,INPUT ,FIXED 192 ::
     CALL CLEAR :: INPUT #1:PUM :: GOTO 1420
370 !
380 ! WEGLEGEN
390 !
400 G$="" :: FOR I=1 TO 5 :: G$=G$&GEP$(I)::
     NEXT I :: IF G$="" THEN DISPLAY AT(12,1):
    "DU HAST KEIN GEPAECK" :: GOTO 2170
410 DISPLAY AT(2,10)BEEP SIZE(1):
    CHR$(62):: ZEI=2 :: G$="" :: FOR I=1 TO 3 ::
     G$=G$&RGE$(I):: NEXT I
420 CALL KEY(3,KEY,STATUS):: IF STATUS=0 THEN 420
430 CALL HCHAR(ZEI,12,32):: IF KEY=11 AND ZEI>2
    THEN IF ZEI=8 THEN ZEI=ZEI-2 ELSE ZEI=ZEI-1 ELSE
     IF KEY=10 AND ZEI<ZMAX
    THEN IF ZEI=6 AND(RUC=1 OR G$<>"")
    THEN ZEI=ZEI+2 ELSE ZEI=ZEI+1
440 CALL HCHAR(ZEI,12,62):: IF KEY=15
    THEN 2170 ELSE IF KEY<>13 THEN 420
450 IF ZEI<7 THEN WAS$=GEP$(ZEI-1)
    ELSE WAS$=RGE$(ZEI-7)
460 GOSUB 2360 :: CALL HCHAR(ZEI,12,32)::
     IF NOT(WAS$="KAEFIG" AND SON(SIT,4)=1)THEN 490
470 IF ZEI<7 THEN GEP$(ZEI-1)
    ="KAEFIG MIT VOGEL" ELSE RGE$(ZEI-7)
    ="KAEFIG MIT VOGEL"
480 SON(SIT,4)=0 :: PU=PU+50 :: DISPLAY AT(12,1):
    "DER VOGEL IST IN DEN KAEFIG GEGANGEN." ::
     FOR I=1 TO 300 :: NEXT I :: GOTO 2170
490 IF ZEI<7 THEN GEP$(ZEI-1)="" ELSE RGE$(ZEI-7)=""
500 IF WAS$="RUCKSACK" THEN RUC=0 :: PU=PU-100 ::
     ZMAX=6
510 IF WAS$="" THEN 2170 ELSE IF WAS$="ZAUBERSTAB"
    THEN PU=PU-13 :: ZS=0 ::
     GOTO 620 ELSE IF WAS$=L$(SIT,4)
    THEN PU=PU+26 :: SON(SIT,1)=0 ::
     GOTO 620 ELSE IF NOT(WU=1 AND SON(SIT,2)=1)
    THEN 620
520 !
530 ! WERFEN
540 !
550 IF WAS$="MESSER" OR WAS$="SPEER" OR WAS$="AXT"
     OR WAS$="STEIN" THEN 570
560 DISPLAY AT(12,1):"DER ZWERG LACHT DICH AUS !" ::
     PU=PU-6 :: GOTO 610
570 WD=INT(2*RND):: IF WD=1 THEN DISPLAY AT(12,1):
    "DU HAST IHN NICHT GETROFFEN!" ELSE DISPLAY
     AT(12,1):"DER ZWERG IST IN EINER
     WOLKEVERSCHWUNDEN..." :: SON(SIT,2),Z=0 ::
     PU=PU+20
580 !
590 ! REAKTION
600 !
610 FOR I=1 TO 300 :: NEXT I
620 IF WAS$="KAEFIG MIT VOGEL"
    THEN WAS$="KAEFIG" :: SON(SIT,4)=1
630 IF SON(SIT,3)=1 AND SON(SIT,4)=1
    THEN DISPLAY AT(12,1):
    "DAS LIED DES VOGELS HAT DEN DRACHEN
     VERTRIEBEN." ELSE 650
640 FOR I=1 TO 400 :: NEXT I :: PU=PU+570 ::
     SON(SIT,3)=0 :: GOTO 660
650 IF WAS$="LAMPE" THEN PU=PU-30 :: AN,LI=0 ::
     CALL HCHAR(16,10,32)
660 FOR I=1 TO 3 :: IF L$(SIT,I)=""
    THEN L$(SIT,I)=WAS$ :: GOTO 690
670 NEXT I
680 DISPLAY AT(12,1):"1 ";WAS$:
    "IST IN EINE BODENSPALTE
      GEFALLEN UND FUER IMMER     VERLOREN!" ::
     GOTO 2170
690 GOSUB 2340 :: DISPLAY AT(12,2):"OK" :: GOTO 2170
700 !
710 ! AUFNEHMEN
720 !
730 G$="" :: FOR I=1 TO 3 :: G$=G$&L$(SIT,I)::
     NEXT I :: IF G$="" THEN DISPLAY AT(12,1):
    "ES GIBT NICHTS, WAS DU{6*(SPACE)}
    MITNEHMEN KANNST." :: PU=PU-2 :: GOTO 2170
740 DISPLAY AT(22,11)SIZE(1)BEEP:CHR$(62):: ZEI=22
750 CALL KEY(3,KEY,STATUS):: IF STATUS=0
    THEN 750 ELSE CALL HCHAR(ZEI,13,32)::
     IF KEY=11 AND ZEI>22
    THEN ZEI=ZEI-1 ELSE IF KEY=10 AND ZEI<24
    THEN ZEI=ZEI+1
760 CALL HCHAR(ZEI,13,62):: IF KEY=15
    THEN 2170 ELSE IF KEY<>13 THEN 750
770 IF L$(SIT,ZEI-21)="" THEN 2170 ELSE
     WAS$=L$(SIT,ZEI-21)
780 FOR I=1 TO 5
790 IF GEP$(I)="" THEN GEP$(I)=WAS$ ::
     L$(SIT,ZEI-21)="" :: GOTO 860
800 NEXT I
810 IF RUC=0 THEN 850
820 FOR I=1 TO 3
830 IF RGE$(I)="" THEN RGE$(I)=WAS$ ::
     L$(SIT,ZEI-21)="" :: GOTO 860
840 NEXT I
850 DISPLAY AT(12,1):"DU KANNST NICHT MEHR
     TRAGEN.WENN DU ETWAS AUFNEHMEN
      WILLST, MUSST DU ERST ETWAS WEGLEGEN." ::
     PU=PU-1 :: FOR I=1 TO 500 :: NEXT I ::
     GOTO 2170
860 GOSUB 2360 :: IF WAS$=L$(SIT,4)
    THEN SON(SIT,1)=1 :: PU=PU-50
870 IF WAS$="ZAUBERSTAB" THEN DISPLAY AT(12,1):
    "AUF DEM STAB STEHT  '";J$;"'":
    "GESCHRIEBEN..." :: ZS=1 ELSE DISPLAY AT(12,2):
    "OK"
880 IF WAS$="RUCKSACK" THEN RUC=1 :: PU=PU+40 ::
     ZMAX=10 ELSE IF WAS$="LAMPE"
    THEN PU=PU+10 :: LI=1 :: CALL HCHAR(16,10,64)
890 GOTO 2170
900 !
910 ! SITUATION VERLASSEN
920 !
930 FOR I=1 TO 3 :: IF L$(SIT,I)=L$(SIT,4)
    THEN SON(SIT,1)=0
```



```
940 NEXT I :: IF RICH$(SIT,RI)=""
    THEN 1500 ELSE ON SIT GOTO
     950,1000,1050,1100,1150,1200,1250,1300
950 ON RI GOTO 960,970,980,990
960 IF SON(SIT,1)=1 THEN 1380 ELSE SN=1 :: GOTO 2300
970 SN=2 :: GOTO 2300
980 IF SON(SIT,3)=1 THEN 1370 ELSE SN=3 :: GOTO 2300
990 SN=5 :: GOTO 2300
1000 ON RI GOTO 1010,1020,1030,1040
1010 IF SON(SIT,1)=1 THEN 1380 ELSE SN=0 :: GOTO 2300
1020 SN=2 :: GOTO 2300
1030 IF SON(SIT,3)=1 THEN 1370 ELSE SN=3 :: GOTO 2300
1040 SN=6 :: GOTO 2300
1050 ON RI GOTO 1060,1070,1080,1090
1060 SN=0 :: GOTO 2300
1070 IF SON(SIT,1)=1 THEN 1380 ELSE SN=1 :: GOTO 2300
1080 IF SON(SIT,3)=1 THEN 1370 ELSE SN=3 :: GOTO 2300
1090 SN=5 :: GOTO 2300
1100 ON RI GOTO 1110,1120,1130,1140
1110 IF SON(SIT,1)=1 THEN 1380 ELSE SN=0 :: GOTO 2300
1120 SN=2 :: GOTO 2300
1130 IF SON(SIT,3)=1 THEN 1370 ELSE SN=4 :: GOTO 2300
1140 SN=5 :: GOTO 2300
1150 ON RI GOTO 1160,1170,1180,1190
1160 IF SON(SIT,3)=1 THEN 1370 ELSE SN=3 :: GOTO 2300
1170 SN=5 :: GOTO 2300
1180 SN=6 :: GOTO 2300
1190 IF SON(SIT,1)=1 THEN 1380 ELSE SN=7 :: GOTO 2300
1200 ON RI GOTO 1210,1220,1230,1240
1210 IF SON(SIT,1)=1 THEN 1380 ELSE SN=0 :: GOTO 2300
1220 SN=1 :: GOTO 2300
1230 IF SON(SIT,3)=1 THEN 1370 ELSE SN=3 :: GOTO 2300
1240 SN=7 :: GOTO 2300
1250 ON RI GOTO 1260,1270,1280,1290
1260 SN=1 :: GOTO 2300
1270 IF SON(SIT,3)=1 THEN 1370 ELSE SN=4 :: GOTO 2300
1280 IF SON(SIT,1)=1 THEN 1380 ELSE SN=5 :: GOTO 2300
1290 SN=7 :: GOTO 2300
1300 ON RI GOTO 1310,1340,1350,1360
1310 IF SON(SIT,1)=1 THEN 1380 ELSE SN=9 ::
     IF ZIEL=0 THEN 2300
1320 DISPLAY AT(12,1):"DU HAST ES GESCHAFFT!!
    {6*(SPACE)}DU BIST DAHEIM.{13*(SPACE)}
    DEIN PUNKTESTAND: ";PU
1330 CALL KEY(3,KEY,STATUS):: IF STATUS=0
    THEN 1330 ELSE 1850
1340 SN=2 :: GOTO 2300
1350 IF SON(SIT,3)=1 THEN 1370 ELSE SN=4 :: GOTO 2300
1360 SN=6 :: GOTO 2300
1370 DISPLAY AT(12,1):"EIN DRACHE VERSPERRT DEN WEG"
     :: GOTO 1500
1380 DISPLAY AT(12,1):"DA IST EIN PROBLEM..." ::
     PU=PU-1 :: GOTO 1500
1390 !
1400 ! BLOCK LADEN
1410 !
1420 BLOCK=BLOCK+1 :: PU=ABS(PU*BLOCK)::
     CALL VCHAR(1,11,124,11)::
     CALL HCHAR(11,1,45,32)::
     CALL HCHAR(17,1,45,32)::
     CALL HCHAR(6,1,45,10):: SIT=1
1430 GOSUB 2340 :: DISPLAY AT(12,1):
    "BLOCK #";BLOCK;"WIRD EINGELESEN.":
    "HIGHSCORE:";PUM :: FOR I=1 TO 8 :: INPUT #1:
    TEXT$(I),COL(I):: NEXT I
1440 FOR I=1 TO 8 :: INPUT #1:
    L$(I,1),L$(I,2),L$(I,3),L$(I,4),SON(I,1),
    SON(I,2),SON(I,3),SON(I,4),SON(I,5):: NEXT I
1450 FOR I=1 TO 6 STEP 2 :: INPUT #1:
    RICH$(I,1),RICH$(I,2),RICH$(I,3),RICH$(I,4),
    RICH$(I+1,1),RICH$(I+1,2),RICH$(I+1,3),
    RICH$(I+1,4):: NEXT I
1460 INPUT #1:RICH$(7,1),RICH$(7,2),RICH$(7,3),
    RICH$(7,4),RICH$(8,1),RICH$(8,2),RICH$(8,3),
    RICH$(8,4),ZIEL :: DISPLAY AT(12,1):
    "VIEL GLUECK!"
1470 !
1480 ! MASKE UND EINGABE
1490 !
1500 IF SON(SIT,5)=1 AND AN=0
    THEN CALL SCREEN(2)ELSE CALL SCREEN(COL(SIT))::
     DISPLAY AT(18,1):TEXT$(SIT)::
     IF SON(SIT,5)=0 AND AN=1 THEN PU=PU-3
1510 IF SON(SIT,4)=1 THEN DISPLAY AT(12,1):
    "EIN VOGEL SINGT EIN LIED.":"":"":""
1520 G$="" :: FOR I=1 TO 3 :: G$=G$&L$(SIT,I)::
     NEXT I :: IF G$="" THEN 1540 ELSE DISPLAY
     AT(22,1)SIZE(10):"DU SIEHST:"
1530 FOR I=1 TO 3 :: DISPLAY AT(21+I,12):
    "1 ";L$(SIT,I):: NEXT I
1540 DISPLAY AT(1,11):"DEIN GEPAECK:"
1550 FOR I=1 TO 5 :: DISPLAY AT(I+1,11):
    GEP$(I):: NEXT I
1560 G$="" :: FOR I=1 TO 3 :: G$=G$&RGE$(I)::
     NEXT I :: IF RUC=0 OR G$="" THEN 1580
1570 DISPLAY AT(7,11):"IM RUCKSACK" ::
     FOR I=1 TO 3 :: DISPLAY AT(I+7,11):
    RGE$(I):: NEXT I :: GOTO 1590
1580 FOR I=7 TO 10 :: DISPLAY AT(I,11):"" :: NEXT I
1590 DISPLAY AT(1,1)SIZE(7):
    "PUNKTE:" :: DISPLAY AT(3,1)SIZE(7):
    PU :: DISPLAY AT(7,1)SIZE(8):
    "WAS TUN?" :: ACCEPT AT(9,1)VALIDATE(UALPHA)
    SIZE(-8)BEEP:TUN$
1600 !
1610 ! AUSWERTUNG DER EINGABE
1620 !
1630 IF TUN$="" THEN 2170 ELSE IF SEG$(TUN$,1,2)
    ="AN" THEN TUN$="AN" ELSE TUN$=SEG$(TUN$,1,3)
1640 GOSUB 2340 :: IF TUN$="WIR" OR TUN$="WER"
    THEN WU=1 ELSE WU=0
1650 FOR BEF=1 TO 8 :: FOR I=1 TO 7
1660 IF BEF$(BEF,I)=TUN$ THEN 1690
1670 NEXT I :: NEXT BEF
1680 DISPLAY AT(12,1):"ICH HABE DEINEN BEFEHL
    {6*(SPACE)}NICHT VERSTANDEN." ::
     FOR I=1 TO 300 :: NEXT I :: GOTO 2170
1690 ON BEF GOTO 1730,1770,1840,1970,1970,1970,1970,
    1970
1700 !
1710 ! HILFEN
1720 !
```

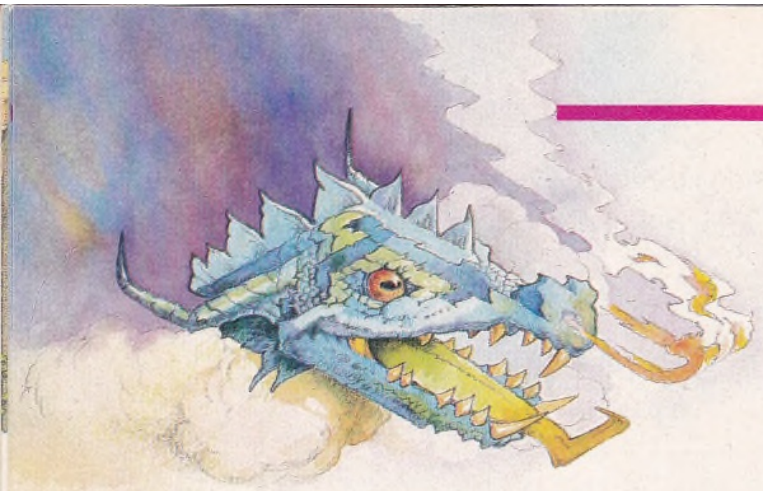


```
1730 IF SON(SIT,1)=1 THEN PU=INT(PU/2)::
      DISPLAY AT(12,1):"DEIN PROBLEM: DU BRAUCHST
       1 ";L$(SIT,4):: GOTO 2170 ::
      ELSE DISPLAY AT(12,1):
     "ES GIBT KEIN PROBLEM." :: GOTO 2170
1740 !
1750 ! LICHT MACHEN
1760 !
1770 IF LI=0 THEN DISPLAY AT(12,1):
     "DU HAST KEINE LAMPE." :: GOTO 2170
1780 IF TUN$="AN" OR TUN$="LIC"
     THEN AN=1 :: CALL HCHAR(16,10,96)::
      IF SON(SIT,5)=1 THEN CALL SCREEN(COL(SIT))::
      PU=PU+10 :: GOTO 2170 ELSE 1800
1790 AN=0 :: CALL HCHAR(16,10,64):: IF SON(SIT,5)=1
     THEN PU=PU-30 :: CALL SCREEN(2)
1800 DISPLAY AT(12,2):"OK" :: GOTO 2170
1810 !
1820 ! AUFGEBEN
1830 !
1840 DISPLAY AT(12,1):"IM ERNST?!?" ::
      ACCEPT AT(12,13)VALIDATE("JN")SIZE(1):
     J$ :: IF J$="N" THEN 2170
1850 CALL CLEAR
1860 !
1870 ! Wahlweise Zeile 1900 fuer Cassetten-
1880 ! oder Zeile 1910 fuer Disketten-Version
1890 !
1900 REM CLOSE #1 :: IF PU>PUM THEN OPEN #1:"CS1",
     INTERNAL,OUPUT,FIXED 192 :: PRINT #1:PU::CLOSE #1
1910 REM CLOSE #1 :: IF PU>PUM THEN OPEN #1:
     "DSK1.HIGH",INTERNAL,OUTPUT :: PRINT #1:
     PU :: CLOSE #1
1920 DISPLAY AT(7,6)ERASE ALL:"REDO FUER NEUEN START"
1930 CALL KEY(4,KEY,STAT):: IF KEY=134
     THEN 270 ELSE IF KEY<>3
     THEN 1930 ELSE DISPLAY AT(11,6):"PUNKTE:";PU:
     TAB(6);"BLOCK :";BLOCK :: STOP
1940 !
1950 ! PECH GEHABT?
1960 !
1970 IF NOT(SON(SIT,5)=1 AND AN=0)
     THEN 1990 ELSE IF SUCH<3
     THEN SUCH=SUCH+1 :: GOTO 2170
1980 CALL SCREEN(7):: DISPLAY AT(12,1):
     "PECH! DU BIST IN EIN LOCH
       GEFALLEN. SOLL ICH DIR{6*(SPACE)}
     HELFEN?" :: ACCEPT AT(14,9)VALIDATE("JN")SIZE(1)
     :J$ :: IF J$="J" THEN 1850 ELSE 1980
1990 ON BEF GOTO 2170,2170,2170,2250,400,730,2030,
     2090
2000 !
2010 ! WAS SAGEN
2020 !
2030 DISPLAY AT(12,1):"WAS SAGEN?" ::
      ACCEPT AT(12,12)VALIDATE(UALPHA):WAS$
2040 IF NOT(WAS$=J$ AND ZS=1)THEN DISPLAY AT(12,1):
     " OK" :: GOTO 2170
2050 GOSUB 2350 :: SN=0 :: DISPLAY AT(12,1):
     "MIT EINEM LAUTEN KNALL{6*(SPACE)}
     VERSCHWINDEST DU..." :: PU=PU-20*BLOCK ::
      GOTO 2300
2060 !
2070 ! FANGEN
2080 !
2090 DISPLAY AT(12,1):"WAS FANGEN?" ::
      ACCEPT AT(12,13)VALIDATE(UALPHA):WAS$
2100 IF WAS$="VOGEL" THEN DISPLAY AT(12,1):
     "PECH! DER VOGEL IST{9*(SPACE)}
     AUFGEFLOGEN." :: PU=PU-5 :: GOTO 2130
2110 IF WAS$="DRACHE" THEN DISPLAY AT(12,1):
     "WIE? ES IST VIEL ZU{9*(SPACE)}
     GEFAEHRLICH!" :: GOTO 2130
2120 IF WAS$="ZWERG" THEN DISPLAY AT(12,1):
     "NU IS A WECH!" :: ZW=0 :: GOTO 1540 ::
      ELSE DISPLAY AT(12,1)BEEP:"QUATSCH!" :: PU=PU-5
2130 FOR I=1 TO 400 :: NEXT I
2140 !
2150 ! EIN ZWERG?
2160 !
2170 IF SON(SIT,2)=0 THEN 1500 ELSE IF Z>5
     THEN DISPLAY AT(12,1):
     "DER ZWERG HAT EIN MESSER
       NACH DIR GEWORFEN UND DICH
       TOEDLICH GETROFFEN!!!":"" ELSE 2190
2180 FOR I=1 TO 300 :: NEXT I :: GOTO 1850
2190 IF Z>2 THEN DISPLAY AT(12,1):
     "DER ZWERG HAT EIN MESSER
       NACH DIR GEWORFEN UND DICH
       KNAPP VERFEHLT!!!":"" :: GOTO 2210
2200 IF Z=0 THEN DISPLAY AT(12,1):
     "EIN ZWERG IST AUFGETAUCHT.": : :
2210 FOR I=1 TO 300 :: NEXT I :: Z=Z+1 :: PU=PU-4 ::
      GOTO 1500
2220 !
2230 ! SICH BEWEGEN
2240 !
2250 G$="" :: FOR I=1 TO 4 :: G$=G$&RICH$(SIT,I)::
      NEXT I :: IF G$="" THEN DISPLAY AT(12,3):
     "KEIN WEG ZU SEHEN." :: PU=PU-2 ELSE 2270
2260 FOR I=1 TO 500 :: NEXT I :: GOTO 2170
2270 FOR I=1 TO 4 :: DISPLAY AT(11+I,3):
     RICH$(SIT,I):: NEXT I :: ZEI=12 :: RI=1 ::
      DISPLAY AT(ZEI,1)SIZE(1)BEEP:CHR$(62)
2280 CALL KEY(3,KEY,STATUS):: IF STATUS=0
     THEN 2280 ELSE CALL HCHAR(ZEI,3,32)::
      IF KEY=11 AND ZEI>12
     THEN ZEI=ZEI-1 ELSE IF KEY=10 AND ZEI<15
     THEN ZEI=ZEI+1
2290 CALL HCHAR(ZEI,3,62):: RI=ZEI-11 :: IF KEY=15
     THEN 2170 ELSE IF KEY=13 THEN 930 ELSE 2280
2300 GOSUB 2350 :: SIT=SN+1 :: IF SIT=10
     THEN 1420 ELSE Z,SUCH=0 :: GOTO 1500
2310 !
2320 ! SUBS MASKEN-FELDER LOESCHEN
2330 !
2340 FOR I=12 TO 15 :: CALL HCHAR(I,1,32,32)::
      NEXT I :: RETURN
2350 FOR I=18 TO 24 :: CALL HCHAR(I,1,32,32)::
      NEXT I :: RETURN
2360 FOR I=22 TO 24 :: CALL HCHAR(I,1,32,32)::
      NEXT I :: RETURN
```

Way Home

Hier nun die Daten, die
das Adventure-Leben
schön und spannend machen!

```
100 CALL CLEAR
110 REM Fuer Diskette Zeile 130
120 REM Fuer Cassette Zeile 140
130 REM OPEN #1:"DSK1.HIGH",INTERNAL,OUTPUT ::
     PRINT #1:0 :: CLOSE #1 :: OPEN #1:
    "DSK1.ADDAT",INTERNAL,OUTPUT,FIXED 192
140 REM OPEN #1:"CS1",INTERNAL,OUTPUT,FIXED 192 ::
     PRINT #1:0
150 FOR BLOCK=1 TO 6 :: PRINT : :BLOCK
160 FOR I=1 TO 8 :: PRINT I;
    :: READ T$,A :: PRINT #1:T$,A :: NEXT I
170 FOR I=9 TO 16 :: PRINT I;
    :: READ A$,B$,C$,D$,A,B,C,D,E :: PRINT #1:
    A$,B$,C$,D$,A,B,C,D,E :: NEXT I
180 FOR I=17 TO 19 :: PRINT I;
    :: READ A$,B$,C$,D$,E$,F$,G$,H$ :: PRINT #1:
    A$,B$,C$,D$,E$,F$,G$,H$ :: NEXT I
190 PRINT I;:: READ A$,B$,C$,D$,E$,F$,G$,H$,A ::
     PRINT #1:A$,B$,C$,D$,E$,F$,G$,H$,A
200 NEXT BLOCK
210 CLOSE #1 :: END
220 !
230 ! BLOCK 1
240 !
250 DATA "DU STEHST VOR EINER HUETTE. EIN PFAD ENDET
     VOR IHR.      HINTER IHR ENTSPRINGT EIN
     BACH.",4
260 DATA "DU BIST IN DER HUETTE. SIE
     HAT NUR EINEN RAUM",8,"DU STEHST AUF EINEM
     HUEGEL. EIN PFAD FUEHRT HINAB.",4
270 DATA "DU BIST AM WALDRAND. EIN WEGUND DER BACH
     FUEHREN IN VER-SCHIEDENE RICHTUNGEN IN DEN WALD
     HINEIN.",4
280 DATA "IM BACHUFER IST EINE HOEHLE.NEBEN IHR
     BEGINNT EIN WILD- WECHSEL. UNTEN FAELLT DER
     BACH UEBER EINE KLIPPE.",3
290 DATA "DER WEG IST SCHON LANGE
     NICHT MEHR ZU ERKENNEN.
     DU HAST DICH VERLAUFEN.",13
300 DATA "DER WASSERFALL VOR DIR IST
     UNBEGEHBAR. WEIT HINTER DIR IST DIE HOEHLE ZU
     SEHEN. IM WALD SCHIMMERT EIN LICHT.",13
310 DATA "DU BIST IN DER HOEHLE. VOR
     DIR EINE TUER. SIE IST ZU.
     DRAUSSEN FLIESST DER BACH.",5
320 DATA "SCHLUESSEL","","","",0,0,0,0,0,"KAEFIG",
    "LAMPE","BROT","",0,0,0,0,0,"","","","",0,0,0,0,
    0
330 DATA "ZAUBERSTAB","MESSER","","",0,0,0,0,0,"",
    "","","",0,0,0,0,0,"","","","",0,1,0,0,0
340 DATA "","","","",0,0,0,0,0,"","","",
    "SCHLUESSEL",1,0,0,0,0
350 DATA "REIN","PFAD","BACH","","RAUS","","","",
    "PFAD","","","","HUETTE","","BACH","WEG"
360 DATA "WALDRAND","WILDWECHSEL","FAELLE","HOEHLE",
    "","","","","","HOEHLE","LICHT","","TUER","",
    "BACH","",0
370 !
380 ! BLOCK 2
390 !
400 DATA "ES IST DUNKEL. VON LINKS
     HOEHRST DU EIN RAUNEN.{6*(SPACE)}
    RECHTS IST EIN HELLER STREI-FEN AUF DEM
     BODEN.",5
410 DATA "DU BIST IN EINEN HELLEN RAUMGELANGT. AUF
     DER EINEN SEITEIST EINE STEILE TREPPE.",4
420 DATA "DU BIST IN EINE STINKENDE
     GRUBE GEFALLEN. HOCH UEBER
     DIR IST EINE TUER. ALLERLEI GETIER IST UM DICH
     HERUM.",5
430 DATA "DU BIST IN EINEN GROSSEN
     RAUM GELANGT. DURCH EINE
     GLASTUER SIEHST DU EIN ANDE-RES ZIMMER.",4
440 DATA "DURCH EINE ANGELEHNTE TUER
     HOERST DU EIN STOEHNEN UND  JAMMERN.",9
450 DATA "DIE RUTSCHE IST VOLL SCHMIERSEIFE. MIT DER
     LINKEN HAST  DU EINEN HALT GEFUNDEN.
     UNTEN IST ES DUNKEL...",5
460 DATA "EINE BREITE RUTSCHE LAEDT
     DICH EIN. LINKS IST EINE
     ENGE WENDELTREPPE. GEGENUE- BER IST EIN
     KAMIN.",4
470 DATA "DURCH DEN KAMIN BIST DU IN
     EIN KLEINES ZIMMER MIT EINERWEISSEN TUER UND
     EINER ROTENTUER GEKOMMEN.",7
480 DATA "LEITER","","","",0,0,0,0,1,"GLASFIGUR",
    "SCHLUESSEL, ROT","","",0,0,0,0,0
490 DATA "TOTE SCHLANGE","","","LEITER",1,0,0,0,1,
    "HOLZLATTE","SCHLUESSEL","","",0,0,0,0,0
500 DATA "","","","",0,0,0,0,0,"","","","",0,0,0,0,
    0,"","","","GLASFIGUR",1,0,0,0,0
510 DATA "","","","SCHLUESSEL",1,0,0,0,0,"STREIFEN",
    "RAUNEN","","","","TUER","TREPPE",""
520 DATA "","TUER","","","","TREPPE","ZIMMER","",
    "RAUM","","TUER","","DUNKEL","","",""
530 DATA "WENDELTREPPE","TUER","RUTSCHE","KAMIN",
    "ROTE TUER","WEISSE TUER","","KAMIN",0
540 !
550 ! BLOCK 3
560 !
570 DATA "IM FAHLEN LICHT KANNST DU
     EINEN ENGEN GANG SEHEN. ER
     BEGINNT VOR EINER VERSCHLOS-SENEN TUER.",7
580 DATA "DU BIST IN EINER TROPFSTEIN-HOEHLE. EIN
     HANG FUEHRT     HINAUS.",10
590 DATA "EINE HOHE CAVERNE OEFFNET
     SICH VOR DIR. EIN STEILER
     WEG GEHT IN EIN LOCH. EIN
     KRIECHGANG BEGINNT RECHTS.",10
600 DATA "ES IST DUNKEL. MAN HOERT
     EINEN FLUSS RAUSCHEN. ENTGE-GENGESETZT IST EIN
     SCHWACHERSCHIMMER ZU SEHEN.",5
610 DATA "DU SIEHST EINEN SEHR TIEFEN UNTERIRDISCHEN
     FLUSS. WEIT  HINTEN ERKENNST DU UEBER IHMEIN
     SCHWACHES LEUCHTEN.",5
620 DATA "DER STEILE WEG ZERFAELLT
     HINTER DIR UND WIRD UN-
     PASSIERBAR. GANZ UNTEN IST
     EINE TROPFSTEINHOEHLE.",9
630 DATA "DU BIST AUF HALBER HOEHE DESHANGES. UNTEN
     SIEHST DU EINETROPFSTEINHOEHLE.",9
640 DATA "DU BIST IN EINEM CANYON AN- GELANGT. HOCH
     UEBER DIR IST EINE PLATTFORM AM FELSEN.",10
650 DATA "RUCKSACK","ROHR","","",0,0,0,0,0,"LANGE
     STANGE","","","",0,1,0,0,0,"","","","",0,0,0,1,
    0,"","","","",0,0,0,0,1
```



```
660 DATA "GOLDKLUMPEN","","","ROHR",1,1,0,0,0,
    "SPEER","","","",0,0,0,0,0,"LEERES
     HEFT","PAPPKARTON","","",0,0,0,0,0
670 DATA "","","","LANGE STANGE",1,1,0,0,0,"",
    "GANG","","","","","","HANG","TUER","",
    "KRIECHGANG","LOCH","","SCHIMMER","FLUSS",""
680 DATA "KRIECHGANG","","","LEUCHTEN","",
    "TROPFSTEINHOEHLE","","","TROPFSTEINHOEHLE","",
    "","HINAUF","PLATTFORM","","FLUSS","HANG",0
690 !
700 ! BLOCK 4
710 !
720 DATA "LINKS IST EINE NIESCHE IM
      FELS. RECHTS BEGINNT EIN
      SANFTER ABHANG ZWISCHEN HO- HEN
     FELSWAENDEN.",9
730 DATA "NEBEN DIR AN DEM SCHMALEN
      FLUSSUFER LIEGT EIN KLEINES BOOT. IM FLUSS IST
     EINE     INSEL ZU SEHEN.",6
740 DATA "DU STEHST AUF EINER SCHMALENFELSKANTE
     MITTEN IM HANG.   TIEF UNTEN IST EIN FLUSS ZU
     SEHEN.",15
750 DATA "EIN ABGRUND BEENDET DEN WEG.EINE
     KOERPERLAENGE UNTER DIRIST EIN SCHMALER SIMS AN
     DERFELSWAND.",11
760 DATA "DEIN BOOT IST LECK. DER WEG ZURUECK IST ZU
     WEIT. BLEIBT NUR DAS ANDERE UFER.",8
770 DATA "DU BIST IN DER NIESCHE. AN  DER WAND STEHT
      'LASS ES !'.EINE SELTSAME ATMOSPHAERE
      HERRSCHT...",7
780 DATA "DU PADDELST DEIN BOOT UEBER DEN FLUSS.
     LANGSAM KOMMT DIEINSEL NAEHER.",8
790 DATA "MIT KNAPPER NOT HAST DU DAS ANDERE UFER
     ERREICHT. EIN   KOBOLD STEHT DORT. HINTER
      IHM IST EIN TOR IM ZAUN.",4
800 DATA "LUFTPUMPE","","","",0,1,1,0,0,
    "TASCHENUHR","","","",0,0,0,0,0,"TRUHE
     GOLD","FLOETE","","SEIL",1,1,0,0,0
810 DATA "","","","",0,0,0,0,0,"STEMMEISEN","","",
    "",0,0,0,0,0,"SEIL","DIAMANTEN","","DIAMANTEN",
    1,1,0,0,0
820 DATA "","","","",0,1,0,0,0,"SPIEGEL","AXT","",
    "FLOETE",1,0,0,0,0,"","","RECHTS","LINKS","",
    "FELSSIMS","","INSEL"
830 DATA "","FLUSS","ZURUECK","","ZURUECK",
    "FELSSIMS","","","","","","ANDERES
     UFER","PLATTFORM","","","",""
840 DATA "ZURUECK","INSEL","","","TOR","","","",0
850 !
860 ! BLOCK 5
870 !
880 DATA "RECHTS NEBEN EINEM GESCHLOS-SENEN TOR IST
     EIN WALD. IN  EINIGER ENTFERNUNG STEHT
     EINEINZELNER BAUM.",4
890 DATA "DU BIST AUF EINEM SCHMALEN
      FREIGEHAUENEN PFAD. DURCH
      DIE BAEUME HINDURCH IST EINELICHTUNG ZU
     ERKENNEN.",13
900 DATA "DU STEHST VOR EINER{9*(SPACE)}
    VERSCHLOSSENEN HUETTE MITTENIM WALD. DU HAST DEN
     WEG    VERLOREN...",13
910 DATA "WEIT IN DER FERNE ZUR LINKENIST EIN
     HUEGEL. RECHTS IST  EIN DICHTER WALD. EIN
     SCHMA-LER WEG FUEHRT HINEIN.",4
920 DATA "DU SIEHST EIN DORF, EINE
      WIESE, EINEN BAUM. EINE
      BODENSPALTE VERSPERRT DEN   WEG ZUM DORF.",4
930 DATA "MITTEN IN DER GRUENEN WIESE VERSPERRT DIR
     EIN FELD VER- DORRTER DISTELN DEN WEG. IN DER
     FERNE IST ETWAS...",11
940 DATA "DER KOBOLD IST WIEDER DA. ERLAESST DICH
     NICHT ZUR WIESE.DU HAST IHM EINMAL ETWAS
      GEGEBEN...",12
950 DATA "DU BIST AM RANDE DEINES
      DORFES. WILLST DU DEN LEUTENALLES ZEIGEN?? DU
     KANNST    DEIN HEIM ERKENNEN!",4
960 DATA "MESSER","FEUERZEUG","","AXT",1,0,0,0,0,"",
    "","","","",0,1,0,0,0,"","","","",0,0,0,0,0,"","",
    "","",0,1,0,0,0
970 DATA "","","","BRETT",1,1,0,0,0,"","","",
    "FEUERZEUG",1,0,0,0,0,"BRETT","","",
    "TASCHENUHR",1,0,0,0,0
980 DATA "","","","TRUHE GOLD",1,0,0,0,0,"WALD","",
    "BAUM","","TOR","","","LICHTUNG","","","","","",
    "RECHTS","LINKS",""
990 DATA "BAUM","WIESE","","DORF","ETWAS","","","",
    "WALD","","WIESE","","HEIM","","","",0
1000 !
1010 ! BLOCK 6
1020 !
1030 DATA "DU HAST EINEN SCHLAG AUF DENKOPF ERHALTEN.
      DU ERWACHST  IN EINEM RAUM MIT 4 TUEREN. SIE
      SIND NUMMERIERT.",7
1040 DATA "DURCH EINE FALLTUER BIST DU IN EIN
      FEUCHTES VERLIES    GESTUERZT.",5
1050 DATA "DU BIST IN EINEM KLEINEN    INNENHOF. EIN
     TORBOGEN UND  ZWEI TUEREN SIND ZU SEHEN.  EINE
     IST VERSPERRT.",10
1060 DATA "DU BIST IN EINEM LANGEN GANGMIT JE EINER
     TUER AN DEN EN-DEN. DIE EINE GEHT AUF EINENHOF.
      DIE ANDERE ...",9
1070 DATA "TIEF UNTEN IST DEIN HAUS AM HANG ZU SEHEN.
      ZWEI TUEREN  UND EIN BRETT FUEHREN VON
      DEM BALKON WEG.",10
1080 DATA "DU BIST IN EINER KAMMER MIT NUR EINEM
     AUSGANG:{10*(SPACE)}EINE KLAPPE IM BODEN.",9
1090 DATA "DU STEHST AUF EINEM FENSTER-SIMS. DER
     KOBOLD HAT DAS    BRETT WEGGESTOSSEN. ER WILL
     BROT HABEN.",14
1100 DATA "EINE SCHRANKE VERSPERRT DEN WEG INS HAUS.
      AUF EINEM     SCHILD STEHT: 'EINWURF DM 1'",4
1110 DATA "MARK","","","MESSER",1,0,0,0,0
1120 DATA "","","","",0,0,0,0,0,"","","","",0,0,0,0,
    0,"","","","",0,0,0,0,0,"","","","STRICKLEITER",
    1,0,0,0,0
1130 DATA "STRICKLEITER","","","",0,0,0,0,0,"","","",
    "BROT",1,0,0,0,0,"","","","MARK",1,0,0,0,0
1140 DATA "TUER 1","TUER 2","TUER 3","TUER
      4","","","","","","TORBOGEN","TUER","","",
    "HOF","TUER",""
1150 DATA "TUER 1","TUER 2","BRETT","HAUS","",
    "KLAPPE","","","","","REIN","","NACH
     HAUSE","","","",1
```

Der direkte Draht

Mit Maschinensprache an's Video-RAM

Beim TI 99/4A sind alle für die Bildschirmdarstellung interessanten Werte im Video-RAM gespeichert. Im Gegensatz zum Mini-Memory und zum Editor/Assembler-Modul bietet das Extended-BASIC keine direkte Zugriffsmöglichkeit zum Video-RAM. Wollen wir auch unter Extended-BASIC Routinen wie CALL POKEV und CALL PEEKV nutzen, müssen wir diese Befehle in Maschinensprache programmieren.

Jedes Modul, mit dem beim TI 99/4A Maschinensprache ausgeführt werden kann, stellt eine ganze Anzahl Hilfsroutinen zur Verfügung. Diese sind in unserem Programm "PEEKV und POKEV für X-BASIC" auch reichlich genutzt, so daß die Routinen mit insgesamt 178 Bytes Länge recht kurz ausfallen. Für die Maschinenprogrammierer unter Ihnen noch ein kurzer Hinweis zu den Hilfsroutinen: Vielfach wird behauptet, daß XMLLNK (ruft Maschinenprogramme auf) im Extended Basic fehlerhaft sei. Das stimmt nicht; es funktioniert lediglich anders als beim Mini-Memory oder Editor/Assembler. Wer sich an die Angaben zum X-Basic im Handbuch zum Editor/Assembler hält, kann nichts falsch machen.

Da wohl nur wenige unter Ihnen ein Diskettenlaufwerk besitzen und Zugriff auf ein Editor/Assembler-Modul haben, können wir die Erklärung des Maschinenprogramms kurz halten: Es werden einige Hilfsroutinen benutzt. NUMREF holt den Wert einer numerischen Variablen, die in CALL LINK aufgeführt wird, aus dem BASIC-Programm in FAC. Das ist ein Zeichenspeicher, der häufig von Routinen des Betriebssystems genutzt wird. NUMASG macht genau das Gegenteil: Es übergibt einen Wert (8 Byte Fließkomma) an eine Variable. CFI rechnet eine Fließkommazahl in einen Integer-Wert um, und CIF rechnet eine Integer-Zahl in eine 8 Byte Fließkomma Zahl um. Diese beiden Routinen werden über XMLLNK aufgerufen. VMBW und VMBR sind Hilfsroutinen, um aus dem Assemblerprogramm in das VDP-RAM zu schreiben oder aus dem VDP-RAM zu lesen. PEEKV und POKEV beginnen jeweils mit dem Laden des Workspace-Registersatzes und dem Holen der Adresse (1. numerischer Wert im CALL LINK-Statement). Dann wird die gesamte Anzahl der Werte von >8312 herbeigeschafft. Dieser Wert Minus 1 (Adresse) ergibt dann die Anzahl der Variablen. PEEKV holt nun die entsprechende Anzahl Bytes aus dem VDP-Ram, rechnet sie in Fließkommazahlen um und übergibt diese an die entsprechenden Variablen. POKEV besorgt erst die Werte der Variablen bzw. die direkt angegebenen numerischen Werte, rechnet sie in Integerzahlen um und schreibt diese ins VDP-RAM.

Aber auch wenn Sie 'nur' eine Speichererweiterung zur Verfügung haben, brauchen Sie auf PEEKV und POKEV für X-Basic nicht zu verzichten. Das entsprechende Ladeprogramm POKEt in der ersten Schleife das Maschinenprogramm. Dann werden die Namen "POKEV" und "PEEKV" mit den Startadressen in die Definitionstabelle gePOKEt, und zu guter Letzt werden noch die Pointer auf den freien RAMbereich geändert. Wenn Sie das Programm eingetippt haben, starten Sie mit RUN 10000. Kommt die Meldung o.k., können Sie alle Zeichen über 400 löschen und das Programm abspeichern.

Andernfalls müssen die Data-Zeilen nochmals überprüft werden. Ist alles in Ordnung, wird CALL INIT eingegeben und dann RUN. Nach Ablauf des Programms sind die Routinen geladen und können angesprochen werden. Sie sind solange im Speicher enthalten, bis einmal das Titelbild angewählt oder wieder CALL INIT durchgeführt wurde.

Ins rechte Bild gesetzt

Die Routinen arbeiten analog dem CALL PEEKV und CALL POKEV des Mini-Memory oder Editor/Assembler. Der Aufruf aus dem X-BASIC erfolgt durch CALL LINK ("POKEV", 161,A,B); mit bis zu 15 Variablen. Um also ein C auf den Bildschirm zu schreiben, geben wir CALL LINK ("POKEV",400,163) ein. 400 ist dabei die Adresse im VDP-RAM und 163 der ASCII-Wert 67 (für C) mit einem Offset von 96. Wir werden diese neuen Befehle aber nun ja nicht für PRINT-Anweisungen verwenden. Besser ist da schon der direkte Zugriff auf die Farbinformationen. Sie stehen unter X-Basic im VDP-Ram ab 2063 (783, die Werte in Klammern beziehen sich auf Mini-Memory und Editor/Assembler) für den Satz 0 (Cursor) bis 2077 für Zeichensatz 14 (799 für Satz 16). Die Werte für die Farben setzen sich zusammen aus: Vordergrundfarbe (Code im Basic) minus 1, das Ganze mal 16, plus Hintergrundfarbe minus 1. Die Farbinformationen werden aber bei jedem Programmende wieder auf Standardwerte zurückgesetzt, ebenso nach der Durchführung eines Statements im Direktmodus. Deshalb ist der Zugriff auf die Farben nur im Programm sinnvoll.

Bei den Sprites gilt, was schon in der letzten Ausgabe in bezug auf das Mini-Memory gesagt wurde. Auch die Adressen sind dieselben. Ab 768 stehen die Informationen für Sprite No. 1, je 4 Bytes enthalten die Lage in Y-Richtung (1. Byte), X-Richtung (2. Byte), ASCII-Charakter mit Offset (3. Byte) und die Farbe (4. Byte), dann folgt der nächste Sprite mit der No. 2. Ist ein Sprite nicht definiert, ist die Y-Lage 192 und die drei weiteren Bytes sind 0. Der ominöse Wert 208, der die Abbildung weiterer Sprites unterbindet, findet sich erst beim 29. Sprite, den es in X-Basic ja nicht mehr gibt (der Video-Display Prozessor könnte aber 32 Sprites). Auch die Geschwindigkeitsinformationen liegen an der gleichen Stelle; ab 1920 befinden sich die ersten vier Bytes für Sprite No. 1. Das 1. Byte ist die Geschwindigkeit in Y-Richtung, das 2. Byte diese in X-Richtung. Byte 3 und 4 werden vom Betriebssystem bei der Errechnung der neuen Lage bei jedem Interrupt genutzt.

Der Zugriff auf das VDP-Ram hat aber auch seine Risiken. In den Bereich 880 bis 1008 sollte nicht gePOKEt werden. Wenn hier falsche Werte stehen, kann es zu einem Systemabsturz kommen. Für Besitzer eines Diskettensystems sollte auch der Bereich von 13784 bis zum Ende des VDP-Rams bei 16383 tabu sein. Hier ein falscher Wert gePOKEt, und die Diskette ist hin! Auch wird oft behauptet, daß falsche Peeks und Pokes den TI 99/4A beschädigen könnten. Das ist aber nur Gerücht.

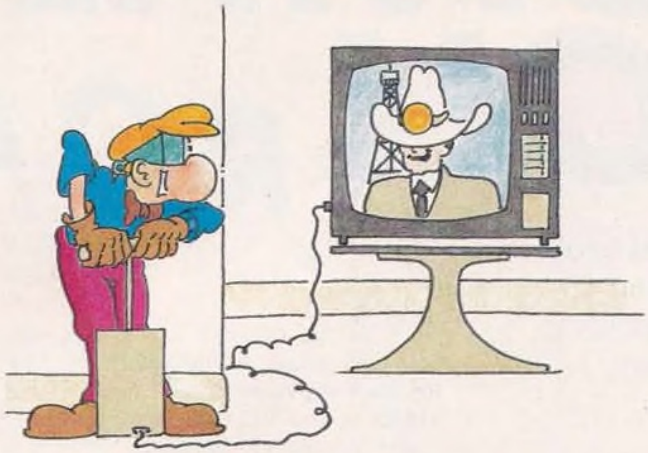
Zum Schluß für heute noch ein kleiner Trick, wie man das Titelbild aus einem Programmablauf heraus erreichen kann. Der Befehl BYE läßt sich ja bekanntlich nicht in einem Programm verwenden. Dazu sind folgende Programmzeilen notwendig (CALL INIT nicht vergessen).

```
10000 CALL PEEK(2,ENDE,ENDE1)
10000 CALL LOAD(-31804,ENDE,ENDE1)
```

Aus dem Betriebssystem holen wir uns (Adresse >0002) die Startadresse der Reset-Routine und poken diesen Wert auf den Zeiger zur anwenderdefinierten Interruptroutine. Beim nächsten Interrupt springt der TI 99/4A dann in den Reset. Dieser 'Befehl' aber ähnelt der Quit-Taste, offene Dateien werden nicht geschlossen, und so kann es zu Datenverlusten kommen.

Das Assembler Listing können Sie mit Freiumschlag bei uns abfordern.

Heiner Martin



```
  100 REM PEEKV UND POKEV ALS DATALISTEN
  110 REM AUFRUF CALL LINK("PEEKV",ADR,VAR,VAR...)
  120 REM UND CALL LINK("POKEV",ADR,VAR,VAR,VAR...)
  130 REM JEWEILS 15 WERTE KOENNEN GESCHRIEBEN ODER
       GELESEN WERDEN
  140 JULY 1984, H.MARTIN
  150 CALL PEEK(8194,C,D,E,F)
  160 IF C*256+B<9461 THEN 190
  170 DISPLAY AT(8,2)ERASE ALL:"ES WIRD EIN PROGRAMM
       UEBERSCHRIEBEN! TROTZDEM LADEN? J/N"
  180 CALL KEY(3,K,S):: IF K=78 THEN END ELSE
       IF K<>74 THEN 180
  190 FOR X=9460 TO 9637
  200 READ A :: CALL LOAD(X,A)
  210 NEXT X
  220 Y=E*256+F-16
  230 FOR X=Y TO Y+15
  240 READ A :: CALL LOAD(X,A)
  250 NEXT X
  260 C=INT(Y/256):: D=Y-C*256
  270 CALL LOAD(8194,37,166,C,D)
  280 PRINT "CALL LINK(""PEEKV"") UND
          CALL LINK(""POKEV"") GELADEN."
  290 END
  300 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
  310 DATA 0,0,0,0,0,0,0,0,0,0,0,0,2,224,36,224,6,
      160,37,148
  320 DATA 192,32,131,74,2,1,36,250,208,160,131,18,9,
      130,6,2,4,32,32,44
  330 DATA 4,192,4,224,131,74,216,34,36,249,131,75,4,
      32,32,24,0,32,192,66
  340 DATA 5,129,4,32,32,8,6,2,22,242,216,0,131,124,2,
      224,131,224,4,96
  350 DATA 0,112,2,224,36,244,6,160,37,148,195,32,131,
      74,4,192,211,96,131,18
  360 DATA 9,141,192,77,6,13,6,160,37,152,216,96,131,
      75,36,248,6,1,2,129
  370 DATA 0,1,22,247,2,1,36,250,192,12,192,141,4,32,
      32,36,4,192,16,221
  380 DATA 2,1,0,1,4,192,4,32,32,12,4,32,32,24,18,184,
      4,91
  390 DATA 80,69,69,75,86,32,37,20,80,79,75,69,86,32,
      37,90
10000 FOR X=1 TO 194
10010 READ A :: B=B+A
10020 NEXT X
10030 IF B=12841 THEN PRINT "O.K." ELSE
       PRINT "DATA-ZEILEN KORRIGIEREN"
```


NEUHEITEN-HITS für COLECO

COMPUTER-SOFTWARE
von Tigervision, Datasoft, Sierra und Hayden im
SONDERANGEBOT
Wir führen auch Software von Ariola — Soft und THORN EMI!

Dukes of Hazzard f. Turbo
Rock'n Rope
Bumb'n Jump
Gateway to Apshai
Keystone Kapers
American Football
Jumpman jr., Tarzan
Star Trek, Decathlon
Frontline f. Act. Contr.
Fortunebuilder
Tally Turtle, Smurf II
Congo Bongo, Puzzle
Tac Scan, Pitfall II

SONDERANGEBOT FÜR VCS 2600
Espial
Polaris
Miner 2049 er
Miner 2049 er II
Springer
Star Trek
Congo Bongo
Buck Rogers
2 Stück DM 65,—*

SONDERANGEBOT FÜR VC 20 + TI 99/4A
von Tigervision + Sega + Sierra on-Line
Steckmodule + Cassetten je DM 59,—*

ZUBEHÖR FÜR COLECO:

Super Action Controler + Rocky	DM 179,—*
Turbo-Module + Turbo	DM 158,—*
Roller-Controler + Slither	DM 179,—*
Atari-Adapter + Wizzard of War	DM 158,—*
Coleco-Konsole + Mouse-Trap	DM 235,—*
Coleco-Konsole + Mouse-Trap	**DM 367,—***

+ Zaxxon + Donkey Kong jr.

SUPER ANGEBOT
GEMINI-Spielkonsole von **CBS** für **ATARI**-Cassetten incl. 2 Joysticks, 2 paddle controler, Antennenumschalter, TV-Anschlußkabel, Transformator, Doppelanschluß (y-Kabel) + Cassette Wizzard of War **DM 240,—***

BIT 90 Heimcomputer
18K + 34K. Der Mikro für die ganze Familie. Dem Vater der Heimcomputer — dem Sohn oder der Tochter das Telespiel! Nicht mehr zwei Geräte, sondern nur noch eines. Voll compatibel zu Coleco Vision Spielcassetten.

Alles sofort lieferbar!

Lieferung von Software aller bekannten Hersteller
* = + Porto u. Verpackung
Wir liefern auch nach Österreich, Schweiz und Benelux.
Fordern Sie unsere kostenlose Preisliste an.

Telefon (0591) 66049

VIDIS-Electronic
Electronic Vertriebs GmbH
Postfach 21 29, 4450 Lingen/Ems, Telex 981137 vidis d
Tel.: (0591) 66049

HORIZONTAL SCROLL

Fein, mittel und grob, ganz nach Wunsch auf beiden Sinclairs zum Einsatz gebracht

Bevor wir Ihnen die Maschinenprogramme zum Scrollen des Bildschrims für beide Sinclairs vorstellen, wollen wir Ihnen erst einmal zum besseren Verständnis den Aufbau des Bildspeichers erklären. Der Bildschirm besteht beim ZX Spectrum aus 176+16 (für den Edit-Bereich) = 192 Zeilen mit jeweils 256 Punkten. Jeder Punkt wird im Bildspeicher durch ein Bit dargestellt, wobei immer acht nebeneinanderliegende Punkte in einem Byte zusammengefaßt sind. Eine Bildschirmzeile bildet im Bildspeicher eine Zeile von 256:8 = 32 Bytes. Nun ist es leider nicht so, daß der Bildschirm Zeile für Zeile von oben nach unten im Bildspeicher abgelegt ist. Um sich zu verdeutlichen, in welcher Reihenfolge sich die Zeilen im Speicher befinden, sollte man das folgende kleine Programm eingeben:

```
100 FOR I = 16384 TO 22527 STEP 32
110 FOR N = O TO 31
120 POKE I + N,255
130 NEXT N
140 NEXT I
```

Da der Bildspeicher sehr groß ist, müssen beim Scrollen des gesamten Bildschirms immer 6144 Bytes bearbeitet werden. Das dauert, wie man sich vorstellen kann, selbst in Maschinensprache recht lange. Daher wird bei den verwendeten Programmen, weil man oft nicht den ganzen Bildschirm zu scrollen braucht, nur jeweils das obere, mittlere und untere Drittel bewegt. Wenn es, wie z.B. bei vielen Spielprogrammen, auf die Geschwindigkeit ankommt, sollte man am besten Grob-Scrolling verwenden. Zum Testen der Maschinenroutinen haben wir rechts ein Demonstrationsprogramm abgedruckt.

Die Assemblerlistings der Programme finden Sie im Kasten auf der folgenden Seite.

... und der ZX 81

Auch auf dem ZX-81 hat man die Möglichkeit, den Bildschirminhalt zu scrollen. Da der ZX-81 keine hochauflösende Graphik hat, verschieben die folgenden Maschinenprogramme den Bildschirminhalt jeweils nur um 1 Zeichen (8Pixels). Ein Scrollen nach links und rechts ist möglich. Beide MC-Routinen sind zusammen 44 Bytes lang und werden in einer REM-Zeile untergebracht.

Also müssen wir zuerst eine REM-Zeile mit 44 beliebigen Zeichen eingeben. Um diese Zeile vor dem unbeabsichtigten Löschen zu schützen, geben wir folgendes ein: POKE 16509,0 und POKE 165100. Unsere REM-Zeile hat nun die Zeilennummer 0 und kann jetzt nicht mehr editiert werden. Als nächstes geben wir ganz einfach folgendes kleine Hilfsprogramm ein:

REMPOKER:

```
10 FOR A = 16514 TO 16557
20 SCROLL
30 PRINT A;" = ";
40 INPUT B
50 PRINT B
60 POKE A,B
70 NEXT A
```

Nachdem wir den REMPOKER mit RUN gestartet und alle Zahlen aus dem MC-Listing eingegeben haben, können wir die Zeilen 10—70 löschen. Nun dürfte nur noch die Zeile 0 mit dem Maschinenprogramm im Speicher sein. SCROLL nach links wird mit RAND USR 16514 und SCROLL nach rechts wird mit RAND USR 16534 gestartet. Um zu überprüfen, ob wir keinen Fehler gemacht haben, geben wir nun folgendes DEMO-Programm ein und starten es anschließend mit RUN:

DEMO

```
10 PRINT AT 10,1;"SCROLL-DEMO"
20 FOR A = 1 TO 19
30 RAND USR 16534
40 NEXT A
50 FOR A = 1 TO 19
60 RAND USR 16514
70 NEXT A
80 GOTO 20
```

Wenn Sie alles richtig gemacht haben, müßte "SCROLL-DEMO" auf dem Bildschirm von links nach rechts und umgekehrt scrollen.

Jens Fretwurst/B.R.

```
  1 CLEAR 29999
 10 PRINT "Scroll"
 20 PRINT "Demo-Programm"
 30 PRINT "? Jens Fretwurst"
100 RESTORE 600
110 FOR i=1 TO 3
120 READ ad,end
130 FOR n=ad TO end
140 READ p: POKE n,p
150 NEXT n: NEXT i
200 LET nn=210: LET usr=30000
210 GO SUB 500
220 LET nn=55: LET usr=30040
230 GO SUB 500
240 LET nn=28: LET usr=30080
250 GO SUB 500
260 PAUSE 100
270 GO TO 200
500 FOR i=8 TO 15
510 PRINT AT i,6;
    "*** Scroll-demo ***"
520 NEXT i
525 PAUSE 50
530 FOR i=1 TO nn
540 LET z=USR usr
550 PAUSE 1
560 NEXT i
570 RETURN
600 DATA 30000,30033
602 DATA 62,1,33,0,64,1,0,8,254
604 DATA 1,32,1,9,254,2,32,2,9
606 DATA 9,6,64,197,6,32,175
608 DATA 203,30,35,16,251,193
610 DATA 16,244,201
650 DATA 30040,30073
652 DATA 62,1,33,0,64,1,0,8,254
654 DATA 1,32,1,9,254,2,32,2,9
656 DATA 9,6,64,197,6,32,175
658 DATA 237,103,35,16,251
660 DATA 193,16,244,201
700 DATA 30080,30122
702 DATA 62,1,33,31,64,1,0,8,254
704 DATA 1,32,1,9,254,2,32,2,9
706 DATA 9,6,64,197,229,229,209
708 DATA 1,31,0,43,237,184,62,0
710 DATA 18,1,32,0,225,9,193,16
712 DATA 235,201
```

MC-Listing: SCROLL nach links

```
16514:  42 12 64 LD HL,(16396)
16517:  14 22    LD C,22
16519:  6 31     LD B,31
16521:  35       INC HL
16522:  35       INC HL
16523:  126      LD A,(HL)
16524:  43       DEC HL
16525:  119      LD (HL),A
16526:  16 249   DJNZ 249
16528:  35       INC HL
16529:  35       INC HL
16530:  13       DEC C
16531:  32 242   JR NZ 242
16533:  201      RET
```

SCROLL nach rechts

```
16534:  42 12 64 LD HL,(16396)
16537:  17 213 2 LD DE,725
16540:  25       ADD HL,DE
16541:  14 22    LD C,22
16543:  6 31     LD,B31
16545:  43       DEC HL
16546:  126      LDA,((HL)
16547:  35       INC HL
16548:  119      LD(HL),A
16549:  43       DEC HL
16500:  16 249   DJNZ 249
16552:  43       DEC HL
16553:  43       DEC HL
16554:  13       DEC C
16555:  32 242   JR NZ 242
16557:  201      RET
```

Grob-Scrolling

Sinclair

```
   LD A,Block    3E xx    wie oben
   LD BC, 2048   01 00 08
   LD HL,16384   21 00 40 fuer linksscroll
  {LD HL,16415   21 1F 40 fuer rechtsscroll}
   CP 1          FE 01
   JR NZ  A1     20 01
   ADD HL,BC     09
A1:CP 2          FE 02
   JR NZ  A2     20 02
   ADD HL,BC     09
   ADD HL,BC     09
A2:LD B,64       06 40
A3:PUSH BC       C5
   PUSH HL       E5
   PUSH HL       E5
   POP DE        D1
   LD BC,31      01 1F 00
   INC HL        23       fuer linksscroll
  {DEC HL        2B       fuer rechtsscroll}
   LDIR          ED B0    fuer linkssroll
  {LDDR          ED B8    fuer rechtsscroll}
   LD A,0        3E 00
   LD (DE),A     12
   LD BC,32      01 20 00
   POP HL        E1
   ADD HL,BC     09
   POP BC        C1
   DJNZ  A3      10 EB
   RET           C9
```

Fein-Scrolling

```
   LD A,Block    3E xx    Block: 0 fuer oberes
                                 1 fuer mittleres
                                 2 fuer unteres
                                   Bilddrittel
   LD HL,16384   21 00 40 fuer rechts-scrolling,
                 21 1F 40 fuer links-scrolling
   LD BC,2048    01 00 08
   CP 1          FE 01
   JR NZ  A1     20 01
   ADD HL,BC     09
A1:CP 2          FE 02
   JR NZ  A2     20 02
   ADD HL,BC     09
   ADD HL,BC     09
A2:LD B,64       06 40
A3:PUSH BC       C5
   LD B,32       06 20
   XOR A         AF
A4:RR (HL)       CB 1E    Feinscroll rechts
  {RL (HL)       CB 16    Feinscroll links}
  {RRD           ED 67    Mittelscroll rechts}
  {RLD           ED 6F    Mittelscroll links}
   INC HL        23       fuer rechtsscroll
  {DEC HL        2B       fuer linksscroll}
   DJNZ  A4      10 FB
   POP BC        C1
   DJNZ  A3      10 F4
   RET           C9
```

PRE-SCAN

Mein TI ist kein Wartesaal!

Das übliche Bild. Man hat ein paar Freunde zu Besuch und will die neuesten, selbstgeschriebenen Programme vorführen. Man lädt, gibt "RUN" ein und wartet. Worauf eigentlich? Was macht der TI da überhaupt solange?

Pre-Scan heißt ins Deutsche übersetzt 'vorprüfen', womit schon einiges erklärt ist. Damit sich das mit "RUN" gestartete Programm nicht auf Betriebssystemebene 'aufhängt', wird vor dem eigentlichen Programmbeginn automatisch eine Syntaxprüfung durchgeführt. Das kann relativ lange dauern: Ein Programm mit 250 Zeilen (ca. 8 k) braucht ungefähr 20 Sek. für diesen Vorgang; das ist bei Programmtests mit häufigen Starts schon recht unangenehm. Nun, diese Wartezeit könnte auf ca. 3 Sek. ohne große Umstände verkürzt werden...

Wie macht man das?

Das Pre-Scan wird durch !αP— ausgeschaltet, d.h. das darauffolgende Statement wird nicht in die Prüfung einbezogen. !αP+ schaltet die Prüfung wieder ein. Man könnte nun ein lauffähiges Programm komplett von der Prüfung ausschließen um die kürzeste Wartezeit zu erzielen.

Das geht aber leider nicht so einfach, denn Bedingung für das Funktionieren von !αP— ist, daß jede im Programm vorkommende Variable und jedes Unterprogramm einmal genannt sein müssen, ehe man dem TI die Prüfung verbietet.

Häufig ist es aber so, daß sich diese kitzligen Stellen auf das gesamte Programm verteilen — in dem Fall ist also alles umsonst? Nein — es gibt da nämlich einen Trick, mit dem man die Pre-Scan-Abschaltung in jedes Programm einbauen kann: Man schreibt am Beginn des Programms alle benutzten Variablen in eine Zeile, schließt diese jedoch vom Programmlauf aus.

Die in Zeile 100 bis 120 dem eigentlichen Programm (Zeile 125 — 160) vorgeschalteten Programm geben nun folgendes Bild:
(α meint dabei stets @)

```
100 GOTO 120
110 A,B,C,D,E,CALL HCHAR
120 !αP—
125 CALL HCHAR C.....
130 IF A=.....
140 LET B=.....
150 C+D=E
160 END
```

Aber Vorsicht!

▷ Wenn die Variablenliste in 110 nicht vollständig ist, dann wird der Befehl mit einem Syntax-Error abgebrochen.

▷ DATA-Zeilen dürfen nicht aus dem Pre-Scan ausgeschlossen werden.

▷ Es wird den meisten TI-Anwendern bekannt sein, daß eine Zeile mit einem IF...THEN Statement innerhalb einer FOR...NEXT Schleife nicht läuft. Dieser Fehler würde, wenn die Zeile nachträglich eingesetzt wird, unentdeckt bleiben; der Befehl wird nicht ausgeführt, das Programm aber auch nicht abgebrochen.

Ronald Früchtnicht

Doppelter Spaß mit doppeltem Joystick

Der Schneider liebt nur Einzelgänger? Wir helfen auf bewährte Art!

Daß es bei jedem neuen Computer, egal welcher Preis- und Leistungs-Klasse, ein Haar in der Suppe der Anwendung oder der Hardware gibt, ist nichts neues.

So auch bei dem von uns getesteten Schneider CPC464 (siehe S.26). Der Rechner präsentierte sich als rundum komplettes System, mit einer Vielzahl von Möglichkeiten, die wohl nur von geübten Programmierern genutzt werden.

Neben einem sehr schnellen und komfortablen BASIC unterstützt der CPC auch alle Schnittstellen, so jedenfalls war es der Anleitung zu entnehmen. Uns befielen doch arge Zweifel. Wir prüften also weiter, und stellten dabei interessante Dinge fest.

Druckerport: Obwohl er etwas seltsam anmutet, ist er voll Centronics-kompatibel. Tests mit Tandy- und Epson-Druckern lieferten einwandfreie Ergebnisse.

System-BUS: Da auf dieser Steckleiste aber auch wirklich alles anliegt, bleibt zu erwarten, daß Lightpen, RS232C, Graphik-Tablett, Disk-Controller und andere Peripherien über kurz oder lang folgen werden.

Stereo-Soundbuchse: Eine witzige Sache, den Sound des Rechners nicht nur über die HiFi-Anlage statt über den eingebauten, aber etwas mickrig geratenen Kleinstlautsprecher, sondern auch noch in Stereo zu hören. Es lohnt sich wirklich!

Joystick-Port: Der Schneider CPC464 ist mit einer Joystick-Buchse ausgerüstet, aber schon der Blick ins BASIC unter dem Befehl JOY belehrt uns, daß zwei dieser Spaß-Bringer angesteuert werden können.

Das gab uns zu denken. Erneutes Wälzen des Handbuches eröffnete, wie das gehen soll; man kaufe den Spezial-Joystick von Schneider, mit Buchse für den zweiten.

Wer aber lieber seine erprobten Kampf-Griffel weiterbenutzen will, hat Pech, denn mehr als ein einziger Atari-kompatibeler zur Zeit ist weder erlaubt noch möglich.

Hier helfen wir jedoch mit unserem Joystick-Adapter-Kabel weiter.

Der Gag mit der einzigen Buchse ist uns ja noch vom TI 99/4 bekannt, so daß wir kurzerhand ein Y-Kabel entwickelt haben, das so einfach ist, daß jeder es nachbauen kann. Man braucht:

— 1 mal Miniatur D 9-pol Stecker female
— 2 mal Miniatur D 9-pol Stecker male
— 8 Standard-Dioden (die ganz billigen reichen)
— einen halben Meter 6-phasiges Kabel

Kosten inclusive Stecker-Gehäuse: kaum mehr als 20 Mark.

Das alles ist in jedem Elektronik-Laden ohne Probleme erhältlich.

Die Dioden bauen Sie sinnvollerweise in die Stecker für die Joysticks (das sind die 'male', die mit den Pins) mit ein. So sind die Dinger elegant aus dem Weg und doch geschützt. In der Buchse für den Rechner-Anschluß (der 'female') wird es etwas eng, da an einigen Anschlüssen zwei Litzen angelötet werden müssen. Saubere Arbeit zwecks Vermeidung von Kurzschlüssen ist nötig.

Zur Belegung:

— Joystickseite:

Pin 1	Links
Pin 2	Rechts
Pin 3	Runter
Pin 4	Hoch
Pin 6	Trigger (Feuer)
Pin 8	Versorgung

— Rechnerseite:

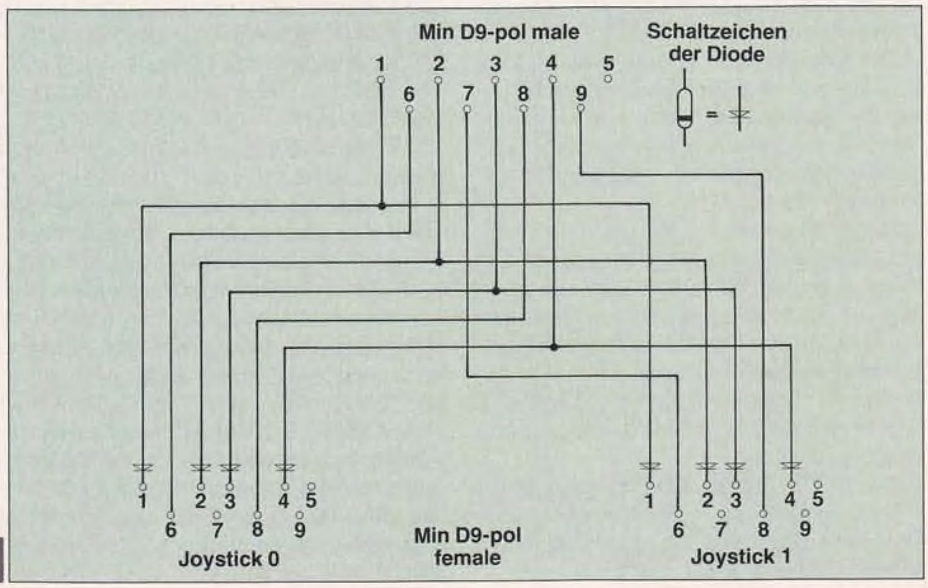
Pin 1	Links
Pin 2	Rechts
Pin 3	Runter
Pin 4	Hoch
Pin 6	Trigger 0
Pin 7	Trigger 1
Pin 8	Versorgung 0
Pin 9	Versorgung 1

Und das ist anders als im Handbuch des Schneider-Rechners. Die Fehler liegen aber nicht bei uns, da man ja einen Atari-kompatiblen Joystick problemlos anschließen kann. Irren ist ja bekanntlich menschlich...

RB

Test-Routine

```
10 CLS
20 PRINT "1 = Hoch":PRINT:
   PRINT "2 = Runter":PRINT:
   PRINT "4 = Links  ; 5 = Li/Ho ;
     6 = Li/Ru":PRINT:
   PRINT "8 = Rechts ; 9 = Re/Ho ;
    10 = Re/Ru":PRINT:
   PRINT "16 = Trigger 0 ;
    32 = Trigger 1":PRINT
30 PRINT ">16 und <27 = Trigger 0 &
    Funktion":PRINT:PRINT ">32 und
    <43 = Trigger 1 & Funktion";
40 LOCATE 10,15:PRINT 0;TAB(19);
   "Stick";TAB(30);1;
50 x=JOY(0):y=JOY(1)
60 LOCATE 10,18:PRINT x;TAB(30);y;
70 GOTO 50
```



Happy X-mas

Nicht nur den TI-Fans wünschen wir wunderschöne Festtage. Doch für sie — stellvertretend für alle unsere Leser — ein elektronischer Gruß!

aus: "Computer total verrückt", Sybex

Es weihnachtet sehr, das läßt sich nun auch am TI nicht mehr verleugnen, auch wenn viele von uns im Rechnerrausch die Festtage beinahe nicht zur Kenntnis genommen hätten. Man hört ja gerade aus den Häusern von frischgebackenen Computerbesitzern die Klagerufe der wohlsorgenden Hausfrauen, deren Festbraten bei weitem nicht die ihm gebührende Beachtung widerfährt, nörgelnde Kinder, die auch man an ihren Computer möchten und Hungerschreie gnadenlos vernachlässigter Kanarienvögel. Muß das so sein?

Wir meinen, nein. Und damit der Haussegen wieder gerade hängt, haben wir uns etwas einfallen lassen. Nutzen Sie also die freie Zeit der Festtage, setzen Sie sich an die Tastatur Ihres texanischen Instruments und füttern Sie den RAM mit Datas.

Natürlich ist bei der Eingabe der Datazeilen mit größter Vorsicht vorzugehen. Nur so kann gewährleistet werden, daß Ihr TI nicht den Weihnachtsmann mit Ihrem Programmresultat auf ewig verschreckt.

Kaum ist nämlich das Programm im Rechner und gestartet, verbreitet sich trotz Plastiktannenbaum und fehlendem Schnee auch schon 1-A-CP/T-geprüfte-Weihnachtsstimmung mit Schallgeschwindigkeit. Eine Routine, die die Kabel Ihres TI zum Schmelzen bringt und den Geruch von Pfeffernüssen im Wohnzimmer diffundiert, mußte leider aus Platzgründen entfallen. Auf Wunsch senden wir Ihnen dieses Listing zu.

Aber Spaß beiseite! Wir wünschen Ihnen recht schöne Tage der Erholung, egal ob vom Computer oder mit Computer.
Ihre CP/T-Redaktion

```
100 CALL CLEAR :: CALL SCREEN(5):: CALL CHAR(113,"
    {8*(0)}1")
110 CALL COLOR(2,16,1,3,16,1,4,16,1,5,4,1,6,15,1,7,
    3,1,8,10,1,9,11,1,10,12,1,11,16,1)
120 Z=5 :: S=15 :: T$="Die" :: GOSUB 760
130 Z=7 :: S=14 :: T$="Musik" :: GOSUB 760
140 Z,S=9 :: T$="Weihnachtskarte" :: GOSUB 760
150 Z=11 :: S=15 :: T$="von" :: GOSUB 760
160 Z=13 :: S=4 :: T$="COMPUTER PRAXIS TELEMATCH" ::
    GOSUB 760
170 FOR I=1 TO 28 :: CALL SPRITE(#I,113,16,1,
    255*RND+1,10,0):: FOR J=1 TO 20 :: NEXT J ::
    NEXT I
180 FOR I=24 TO 1 STEP -1
190 X$="{16*(0)}{16*(F)}" ::
    CALL CHAR(39+I,SEG$(X$,1,16))::
    CALL HCHAR(I,1,39+I,32)
200 FOR J=1 TO 8 :: X$=SEG$(X$,3,LEN(X$))::
    CALL CHAR(39+I,SEG$(X$,1,16)):: NEXT J
210 NEXT I :: CALL SCREEN(16):: CALL CLEAR ::
    CALL DELSPRITE(ALL)
220 FOR I=1 TO 13 :: CALL COLOR(I,2,1):: NEXT I ::
    CALL SCREEN(8)
230 FOR I=33 TO 134 :: READ X$ :: CALL CHAR(I,X$)::
    NEXT I :: CALL SCREEN(6)
240 DATA 8080808080808080,8080808081828488,
    8C8C8E8E86868281,808082848F838684,
    00001060C0808080,04081020,{11*(0)}
    10204,0000040242436538
250 DATA 1021478F3E384080,909040201108O707,
    000306060408101,FF1771892638,1C1C1E3F3F1F8381,
    2619604,0C04000014E81020,020404040814140C
260 DATA {11*(0)}10102,102824247E7FFF7F,
    0505090909090909,7C72E28103050509,{7*(0)}
    11FFCFDFC,F1E080,F1F8FEE7438F357F,
    1C0C0E0F4CBFC7F8
270 DATA {12*(0)}201,0C0402010101,1F383E3F3F3F1C0C,
    010E718384050919,0303060C38E0C080,{12*(0)}
    F03F,106080,031C631C30C1060F
280 DATA F8787C3E01010101,FF8FACF8E0F0F8F8,1F3F7
    {11*(F)},FE7F7F3F3F3F1F1F,{7*(0)}
    20FF4FFFF,8{7*(0)}F2FE0604,800FF9D7785060E0
290 DATA {14*(0)}FF,80E0381E07,{9*(0)}
    71C70C0,07882847800080E0,FF81,{16*(F)}
    ,22221313B7FFFF7F,00000101011F5220,01FCA1DD1D02
300 DATA 6060C0C0C0808080,{6*(0)}
    FF60606060,20204041424C7080,B1A1615050482720,
    030303070B0B1311,0202838282818303,
    1111121202020202
310 DATA D1F1F1F1E1E12222,C038E67151515151,
    0102040810204080,0706060606020201,
    0307070707070707,01033FC101030303,
    0301010102C3481
320 DATA 80800000F0080402,6078E465C2C2C1C1,
    3030303030606F,0C0C0C1818303030,{7*(0)}
    20206060C,0F0F0703030101,C0C06133371F0E0F
330 DATA {6*(0)}8080C0C0C0,78E0E0C080,
    0202010101000071E,0102810140B00C02,
    A050484824240202,0402000080404020,
    0080C0E0F0783F3F,78F0F0E0C080
340 DATA 0103070E0E1C38E8,2F15080402020101,
    1010101010A04,008040A06020201,{7*(0)}
    10306FFFF,08040201,808040402020101,FF55,
    103068C88404FFFF
350 DATA FC5601,0304080810080402,FC0603,
    00C04E4A6040,36C9,00C03C03,0209F21404080804,
    1CEE0E0F0F070703,10080402C1601C07,6CC3,
    8888444422221111
360 DATA 4845C6C44850202,0808844444241C08,
    101020204080808,C060201008040202,{12*(0)}3EC1
370 CALL VCHAR(23,6,33,2)
380 CALL HCHAR(22,6,34)
390 CALL HCHAR(21,6,33)
400 FOR I=0 TO 4 :: CALL HCHAR(20-I,6,35+I):: NEXT I
410 CALL HCHAR(24,7,40)
420 FOR I=0 TO 8 :: CALL HCHAR(22-I,7,41+I):: NEXT I
430 FOR I=0 TO 10 :: CALL HCHAR(24-I,8,50+I)::
    NEXT I
440 FOR I=0 TO 11 :: CALL HCHAR(24-I,9,61+I)::
    NEXT I
450 CALL HCHAR(24,10,73):: CALL HCHAR(23,10,74)::
    CALL HCHAR(21,10,75):: CALL HCHAR(19,10,76)::
    CALL HCHAR(18,10,77)
```

```
460 FOR I=0 TO 3 :: CALL HCHAR(17-I,10,77+I)::
    NEXT I :: CALL HCHAR(13,10,72)
470 CALL VCHAR(23,11,33,2)
480 FOR I=0 TO 8 :: CALL HCHAR(22-I,11,81+I)::
    NEXT I
490 FOR I=0 TO 3 :: CALL HCHAR(24-I,12,90+I)::
    NEXT I
500 CALL HCHAR(20,12,63):: FOR I=0 TO 5 ::
    CALL HCHAR(19-I,12,94+I):: NEXT I
510 FOR I=0 TO 7 :: CALL HCHAR(24-I,13,100+I)::
    NEXT I
520 CALL HCHAR(16,13,72,4):: CALL HCHAR(17,16,107)::
    CALL HCHAR(18,15,107):: CALL HCHAR(19,16,107,2)
530 FOR I=0 TO 5 :: CALL HCHAR(24-I,14,108+I)::
    NEXT I
540 FOR I=0 TO 3 :: CALL HCHAR(24-I,15,114+I)::
    NEXT I :: CALL HCHAR(21,16,117)::
    CALL HCHAR(24,16,118):: CALL HCHAR(23,16,107)
550 CALL HCHAR(24,17,43):: CALL HCHAR(23,17,39)::
    CALL HCHAR(21,17,119):: CALL HCHAR(20,17,120)::
    CALL HCHAR(17,17,121)
560 CALL HCHAR(23,18,69):: CALL HCHAR(21,18,122)::
    CALL HCHAR(20,18,123):: CALL HCHAR(17,18,124)
570 CALL HCHAR(23,19,83):: CALL HCHAR(22,19,43)::
    CALL HCHAR(21,19,125):: CALL HCHAR(20,19,126)::
    CALL HCHAR(17,19,127)
580 CALL HCHAR(23,20,128):: CALL HCHAR(18,20,121)::
    CALL HCHAR(17,20,107)::
    CALL HCHAR(21,20,128,2):: CALL HCHAR(23,21,46)
590 CALL HCHAR(23,22,84):: CALL HCHAR(23,23,113)::
    CALL HCHAR(18,21,127):: CALL HCHAR(17,21,120)::
    CALL HCHAR(16,20,120)
600 CALL HCHAR(15,19,106):: CALL HCHAR(16,20,106)::
    CALL HCHAR(14,21,106,3)::
    CALL HCHAR(15,22,116):: CALL HCHAR(16,22,129,2)
610 CALL HCHAR(15,23,129)::
    CALL HCHAR(17,22,130,2)::
    CALL HCHAR(18,22,131):: CALL HCHAR(18,23,132)
620 CALL HCHAR(19,22,45):: CALL HCHAR(19,23,78)::
    CALL HCHAR(22,23,109):: CALL HCHAR(21,23,133)::
    CALL HCHAR(20,22,134)
630 CALL HCHAR(24,23,67):: CALL SCREEN(5)
640 CALL MAGNIFY(3):: CALL
    CHAR(136,"010103030307FF3F1F0F1F1E1C382{7*(0)}
    808080C0FEF8F0E0F0F0703808")
650 CALL SPRITE(#1,136,12,20,120)
660 FOR I=1 TO 46 :: READ D,T1,T2 ::
    CALL SOUND(D,T1,0,T2,5):: NEXT I ::
    FOR I=1 TO 2000 :: NEXT I
670 RESTORE 700
680 FOR I=1 TO 46 :: READ D,T1,T2 ::
    FOR J=0 TO 50 STEP D/100 ::
    CALL SOUND(-D,T1-J,0,T2-J,5):: NEXT J :: NEXT I
690 RESTORE 700 :: GOTO 660
700 DATA 600,587,589,200,659,661,450,587,589,1100,
    494,496,600,587,589,200,659,661,450,587,589,
    1100,494,496
710 DATA 1000,880,885,450,880,885,1000,740,745,1000,
    784,789,450,784,789,1000,587,589,1000,659,661,
    400,659,661
720 DATA 800,784,789,200,740,745,400,659,661,600,
    587,589,200,659,661,400,587,589,1100,494,496
730 DATA 1000,659,661,400,659,661,800,784,789,200,
    740,745,400,659,661,600,587,589,200,659,661,400,
    587,589
740 DATA 1100,494,496,1000,880,885,400,880,885,800,
    1047,1055,200,880,885,400,740,745,1000,784,789,
    1100,988,996
750 DATA 450,784,789,200,587,589,400,494,496,600,
    587,589,200,523,525,400,440,442,2000,392,395
760 FOR I=1 TO LEN(T$):: CALL
    HCHAR(Z,S+I,ASC(SEG$(T$,I,1))):: NEXT I ::
    RETURN
```


BAUSATZ-FREUDE FÜR ROBOTER-FREUNDE

Diese Roboter sind zum Bauen da. Voll beweglich und, abhängig vom Maßstab, ausbaufähig. Etwa mit Elektromotoren oder mit Leuchtdioden. (Nicht im Lieferumfang enthalten). Richtig zu Weihnachten. Zum Schenken lassen oder Verschenken. Wir liefern sofort!

Best.Nr.	Typ	Maßstab	früherer Preis	jetzt nur
R 1	Gundam	1 : 100*	16,90	**10,90**
R 2	Gundam	1 : 72**	65,—	**55,—**
R 3	Gundam	1 : 60***	56,90	**49,—**
R 4	Zaku	1 : 100	16,90	**10,90**
R 5	Zaku	1 : 72	65,—	**55,—**
R 6	Zaku	1 : 60	56,90	**49,—**
R 7	Domu	1 : 100	18,90	**12,90**
R 8	Gelgoog I	1 : 100	18,90	**12,90**
R 9	Gelgoog II	1 : 100	18,90	**12,90**
R 10	Gok	1 : 100	18,90	**12,90**
R 11	Raumstation	1 : 1200	25,90	**19,90**

Ca. Höhe der fertigen Modelle * = 15 cm; ** = 35 cm; *** = 45 cm

BESTELLCOUPON

------- Ausschneiden und einsenden -------

PORT PRESS Verlag GmbH - Papenhuder Straße 6 - 2000 Hamburg 76

Hiermit bestelle ich zu den untenstehenden Lieferbedingungen

Anzahl	Best.Nr.	Kurztitel	Preis pro Stück	Gesamt

Name _____ Vorname _____

Straße _____

PLZ/Ort _____

+ Versandkosten DM 5,—

Gesamt _____

Den Gesamtbetrag von DM _____ habe ich auf Ihr Postscheckkonto Hamburg 18 79 79-209 (BLZ 200 100 20) überwiesen/ liegt als Verrechnungsscheck anbei. (Zutreffendes bitte unterstreichen) Absender und Anschrift nicht vergessen!

CP/T 12/84

Unsere Lieferungs- und Zahlungsbedingungen: Alle hier angebotenen Artikel sind nur begrenzt lieferbar. Die Auslieferung erfolgt in der Reihenfolge des Bestellungs- bzw. Zahlungseingangs. Bei Bestellung bitte deshalb unbedingt Ersatznummer angeben. **Keine Lieferung per Nachnahme.**

TAKE 1 Walt Disney

PROGRAM MENU

Was der Meister des Zeichentrickfilms mit dem Computer zu tun hat? Ganz einfach: Computer sind, entsprechende Software vorausgesetzt, zum Zeichentrickfilm-Machen, zur Animation da. War das bisher nur in Perfektion mit Mainframes möglich, so gibt es jetzt Möglichkeiten, tatsächlich ebenso schön und glatt wie Walt Disney zu animieren. Es liegt allein bei Ihnen, ob Sie Micky, Donald und Company mit Ihrem Computer zum Leben erwecken wollen ...

Die Kurzpräsentation auf der Sommer-CES, Chicago hatte bereits unseren Animations-Appetit geweckt. Denn: Einen wirklich perfekten Trickfilm auf dem Computer zu einem lächerlichen Preis generieren zu können, war bislang zwar Wunschtraum vieler User, aber eben nicht realisierbar, sehen wir einmal vom "Movie Maker" (bzw. Atari Kintopp) ab, der jedoch in den Möglichkeiten limitiert ist. Folglich waren wir auf das Produkt besonders gespannt. "Klappe auf" also für "Take 1".

Mit der 'Klappe' fängt alles an: Sie ist Bestandteil des Programm-Menüs, das nach dem Booten auf dem Bildschirm erscheint. Unvermittelt marschiert von links ein hutbewehrter Knilch namens Lit'l Dog'n ins Bild, wandert hinter die Klappe und verschwindet kurzzeitig wieder am rechten Bildschirmrand. Nicht schlecht, dieses Minidemo!

Das Menü wartet mit sechs Wahlmöglichkeiten auf, die nachstehend näher erläutert werden. Da haben wir neben "Pictures & Backgrounds" noch "Actors & Actions", dazu den "Scene Editor", einen "Movie Editor", den "Movie Projektor" und schließlich — selbstverständlich — die "Disc Utilities".

Uns interessierte zunächst einmal der Demo-Mode. Über die Cursorsteuerung wählen wir den "Movie Projektor" an, worauf automatisch geladen und dann die Frage nach "Einzelvorführung" (Single Play) oder "Dauervorführung" (Continuous Re-Run) gestellt wird. Zwei Beispiele sind auf der Diskette enthalten, die wir auf diesen Seiten auch zeigen, das Space Shuttle-Demo, sowie das AAA-Industries-Demo. Hier bereits wird deutlich, welch' perfekte Animation mit "Take 1" möglich ist. Bei "Space Shuttle" schwebt eben dieses in die Nähe eines Satelliten, worauf sich die Frachtluken öffnen und anschließend — mit Zoom-Effekt quasi — ein Astronaut aus der Kommandozentrale heraus- und an den Satelliten heranschwebt. Dann ein 'Cut': Wir können die Landung des Shuttle in der Wüste verfolgen.

Das AAA-Industries-Demo verdeutlicht, daß das Programm durchaus zu (semi)professionellen Zwecken, etwa einer animierten Präsentation o.ä. eingesetzt werden kann. Auf dem Schirm wird eine Statistik mit Quartalsumsätzen eines Unternehmens gezeigt; die Zuwachsraten werden im Sinne des Wortes ins Bild via Gabelstapler hineingefahren. Soweit so gut. Aber wie 'filmt' man nun mit "Take 1"?

Das Vorhandensein eines Drehbuchs vorausgesetzt, wie sich's für Filmer nun gehört, wird der "Scene Editor" gewählt. Das Laden erfolgt in wenigen Sekunden; das "Function-Menü" erscheint. Durch Ansteuern von "Shoot Scene" treffen wir bereits die ersten Aufnahmevorbereitungen. Was jetzt benötigt wird, ist ein Darsteller, ein "Actor". Den verpflichten wir durch "Load Actor", wobei es über die drei vorgegebenen Akteure hinaus, darunter auch Titelheld Lit'l Dog'n, unbegrenzte Möglichkeiten gibt. Denn: Mit Hilfe a) des Koala-Touch Pad, b) des Gibson Light Pen Systems, c) des Baudville Programms "Pixit" und d) des Baudville Programms "Blazing Paddles", das wir nachstehend präsentieren, ergänzt um die umfangreiche Baudville-Shape Library, lassen sich Abertausende von Charakteren gestalten.

Bleiben wir aber beim fertigen Angebot. Durch Druck auf RETURN wird unser Actor aktiviert, der Bildschirm leert sich bis auf's Shoot-Menü, mit dem die Frage nach der Frame Rate (Bilder pro Sekunde) erfolgt. Der Standard von 10 Bildern pro Sekunde läßt sich beliebig modifizieren (üblich sind übrigens 24 Bilder pro Sekunde, um einen wirklich flüssigen Animationsablauf zu erzeugen). Der Bildschirm zeigt nun das "Status Window". Da wir ja mit dem Filmen beginnen, wählen wir "ADD", worauf das Prompt "ADD Frame //of 1" erfolgt. Zudem wird rechts der noch freie Speicherplatz angezeigt, der in diesem Stadium bei 7921 Bytes liegt.

Wer jetzt Hilfe braucht und den Blick ins hervorragende, übersichtlich gegliederte Manual scheut, drückt auf "?" und findet dort sämtliche Befehle, die für's Weitermachen erforderlich sind. Durch erneuten Druck auf "?" befinden wir uns wieder bei "SHOOT SCENE". Die Betätigung der Taste "C" bringt unseren Akteur auf den Schirm, umrahmt von einem flimmernden Fenster, mit dem seine Position eindeutig definierbar ist. Im Menü erscheinen nun die Werte X und Y,

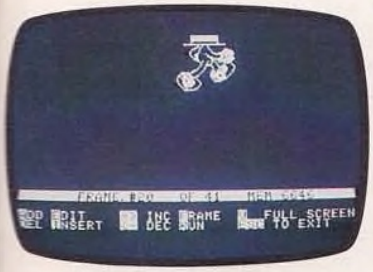
läßt schön grüßen

mit denen die Bildschirmkoordinaten wie üblich festgelegt werden. Mittels Keyboard kann der Akteur nun entweder in Schritten von einem Pixel, zwei oder zehn Pixels an jeder beliebigen Stelle plaziert werden. Dabei entspricht X der Horizontal-, Y der Vertikalposition auf dem Schirm. An die gewünschte Position gebracht, erfolgt die Aufnahme durch Betätigung der "RETURN"-Taste, dies in Relation zur gewünschten Animationszeit. Was bedeutet: Um einen glatten Ablauf zu erzeugen, nimmt man etwa eine Phase nur ein- bis zweimal auf, bewegt den Darsteller weiter und verfährt wie zuvor. Nach Abschluß dieser Sequenz kann (einzelbildweise!) eine Kontrolle erfolgen, kann der Ablauf vor- und rückwärts betrachtet und jederzeit korrigiert werden. Stimmt das Ergebnis, speichert man es auf Diskette.

Und jetzt wird's erst richtig interessant, denn "Take 1" bietet natürlich mehr. So lassen sich etwa über das Menü "Actors & Actions" wiederum sowohl fertig konfektionierte oder selbst kreierte Bilder laden, Ausschnittsvergrößerungen herstellen, über den "Snapshoot"-Mode zwischenspeichern und an beliebiger Stelle einfügen. Dabei bedient man sich einer effektiv einsetzbaren Windowing-Art. Beim "Snapshooten" — das haben wir allerdings noch nicht in Ruhe ausprobieren können — erfolgt die Hauptarbeit im SPRITE-Modus, was über ein normales Sprite hinaus bedeutet, daß man transparente Fenster oder 'Löcher' im Bild nach Belieben einsetzen kann.

Die folgenden Arbeiten sind völlig mit professioneller, konventioneller Animationstechnik identisch: Die gefertigten Szenen werden auf Bewegungsablauf überprüft und, so man will, in der Bewegungsrichtung modifiziert. Das Ergebnis sollte nun wieder gespeichert werden.

Nächster Schritt ist die Erstellung eines Hintergrundes, die mit den vorgenannten Programmen bzw. Peripherals erfolgt. Der Menü-Aufbau stimmt mit dem des "Actors"-Menü weitgehend überein, gesteuert wird wiederum über Keyboard.



Zusammengefügt werden die so erzeugten Elemente folgerichtig im "Scene Editor", über den der Hintergrund in die Szene kommt, Text eingebracht werden kann und — dies ist wichtig — Sequenzen einzelner Akteure quasi zu 'Massenszenen' editiert werden. Die Aufnahmen erfolgen wie gehabt einzelbildweise, was natürlich eine Menge Zeit in Anspruch nimmt. Aber das ist — Computer hin oder her — bei jedem Zeichentrickfilm so. Das i-Tüpfelchen wird dem Produkt durch Hinzufügen des Sounds draufgesetzt, wobei fertig allerdings nur zehn Effekte zur Verfügung stehen. Hier werden sich interessierte User sicher noch zusätzliche Gags einfallen lassen.

Abschließend erfolgt, um das Werk zu vollenden, jene Arbeit, die üblicherweise im Schneideraum getätigt wird, das letzte Editieren des Films mit dem "Movie Editor". Beliebig lassen sich Szenen einfügen, herausschneiden und nochmals modifizieren. Und dann erfolgt die Projektion so, wie einleitend beschrieben.

Ein Traumprogramm steht da zur Verfügung, mit dem wir intensiv weiter arbeiten werden. Wenn man auch Ruhe und Zeit zur Einarbeitung braucht, — die Ergebnisse sind hervorragend. Ein Preis/Leistungsverhältnis, das rundum stimmt. Hoffen wir, daß "Take 1" bald auch für andere Systeme zu haben sein wird.

h.h.

TAKE 1
Hersteller: Baudville
System: Apple II Computer
Programm: Diskette
(Getestet auf Apple II +)

Blazing Paddles

Hersteller: Baudville
System: Apple II +
Programm: Diskette

Auf der Suche nach bedienerfreundlichen, neuartigen Grafikprogrammen stießen wir, eher zufällig, auf "Blazing Paddles", eines von immerhin drei Softwarepaketen (im weiteren Sinne) des Chicagoer Newcomers Baudville. Sicher: Betrachtet man Art, Anordnung und Aufbau des Menüs, scheint die direkte Verwandtschaft zum "Micropainter" und ähnlichen Programmen offensichtlich. Die Extras lassen sich aber unschwer nach wenigen Minuten Umgang mit den "Blazing Paddles" erkennen.

Vier Steuerungsmöglichkeiten sind im Angebot: Ob Maus, Touchpad oder Graphics Tablet — Probleme gibt's hier keine. Anders indes bei der vierten Steuerungsart: Alle verwendeten Joysticks, über zwanzig verschiedene standen uns zur Verfügung, erwiesen sich als völlig ungeeignet, da denkbar unpräzise. Anfangsschwierigkeiten zeigten sich

Blazing Paddles

auch bei der Konfiguration mit dem Koala-Touchpad, erwiesen sich aber schließlich als Eingewöhnungsprobleme.

Wie bei den meisten anderen Grafikprogrammen wird auch die Arbeit bei "Blazing Paddles" durch Pictogramme erleichtert. 20 Möglichkeiten bietet das Hauptmenü, wobei neben den hinlänglich bekannten, die Optionen "Window", "Text" und "Shapes" besonderer Erwähnung bedürfen.

"Window" ist als Synonym für die "Cut & Paste"-Funktion (also Ausschneiden und Einkleben) zu verstehen. Ein beliebiger Teil des wie auch immer erzeugten Bildes kann bei der Arbeit mit der Select Window Option ausgeschnitten und montiert werden. Die Option "Text" läßt die zusätzliche Verwendung von Schriften zu. Bei "Shapes" schließlich bedient man sich einer Reihe vorprogrammierter Elemente, die sich beliebig modifizieren lassen, oder — und hier liegt die entscheidende Besonderheit des Programms — der Anwender verwendet das reichhaltige Angebot der "Shape Library" von Baudeville. Nach unserer Auffassung ist dieses dreiteilige Grafik-Komplettpaket, bestehend aus "Blazing Paddles", "Pixit" und "Take 1", das perfekteste und preiswerteste im derzeitigen Angebot. Wie zuvor festgestellt: Lieferbarkeit für andere populäre System wünschen wir uns schnellstens. *h.h.*

Der Anstreicher

Eine Softwareerweiterung für das TI-Grafikpaket Exbasic II zeigt, was sie kann

Unsere aufmerksamen Leser werden sicherlich mit dem Begriff Exbasic II das Richtige verbinden. Maschinenprogramm und hochauflösende Grafik sind die Stichworte, die hier als Erinnerungsstütze an dieses Programm dienen sollen. Als Ergänzung wird nun von der Firma Computer Hüsli der Painter angeboten, bei dem es sich um ein eigenständiges Programm handelt. Voll in Maschinensprache geschrieben, verspricht Painter einiges.

Nach dem Laden des Programms von Diskette trieft noch keine Farbe aus dem Laufwerk, es geschieht noch gar nichts. Lediglich ein Pfeil auf dem Bildschirm zeigt die Arbeitsbereitschaft des Programms an. Wird auf FCTN 9 gedrückt, zeigt sich in den oberen zwei Bildschirmzeilen das Programmenü, das aus 16 Farben und einigen Pictogrammen besteht.

Mit dem Joystick läßt sich dieser Pfeil auf dem Bildschirm auf irgendein Pictogramm steuern. Durch Betätigung des Feuerknopfs wird die Funktion, für die das Bildchen steht, übernommen. Und dann kann das Painten losgehen.

Vielfältige Möglichkeiten

Man kann Linien ziehen, Kreise und Ellipsen zeichnen, einzelne Punkte setzen, Texte eingeben, Kurvenzüge farbig ausfüllen, den gesamten Bildschirm löschen, wenn ein Entwurf in die Hose gegangen ist, gelungene Zeichnungen auf Diskette speichern oder auch laden oder einzelne Bildpartien ganz gezielt löschen.

Besonders interessant ist die Möglichkeit, über Exbasic II geschaffene Grafiken von Diskette in den Painter zu laden, um ihnen den letzten Schliff zu geben. Der umgekehrte Weg kann natürlich auch gegangen werden. So können durchaus sehr schöne Titelbilder für hausgemachte Spiele entstehen.

Wir können es uns an dieser Stelle sparen, auf die einzelnen Programmfunktionen näher einzugehen, da ihre Funktion und Bedeutung genauso eindeutig und narrensicher sind, wie ihre Bedienung. Ermöglicht wurde dies durch die hervorragend gelöste Menütechnik in Verbindung mit den Pictogrammen. So etwas würden wir gerne öfters sehen. Die gesamte Steuerung des Programms kann vom Joystick aus erfolgen. Wem der Joystick allerdings zu ungenau erscheint, der kann über die Cursortasten die gewünschte Feinjustierung vornehmen.

Der berühmte Schatten

Leider müssen wir am Painter einiges kritisieren. Es lassen sich damit zwar Kreise erzeugen. Allerdings entpuppen sie sich bei genauem Hinsehen als Ellipsen, da zwei Radien definiert werden müssen. Wer aber schafft es schon, den Arbeitspfeil mit dem Joystick z.B.: 90 Pixel in der horizontalen und anschließend 90 Pixel in der vertikalen Richtung auszulenken?

In den meisten Fällen werden zwei unterschiedliche Radien an den Rechner weitergegeben, der dann treu und brav wie immer, eine Ellipse statt eines Kreises zeichnet. Ellipsen sind also kein Problem, aber Kreise...? Bleiben wir bei den Kreisen. Wer einen Kreis mit dem Radius null oder eins zeichnen will, läßt dummerweise seinen geliebten Rechner ins Nirwana wandern. Was das bedeutet, brauchen wir niemandem zu sagen: Powerreset, Verlust aller Daten, Programmneustart. Eine Schwäche des Programms, die man schnell beheben sollte.

Wer das Wort Painter ernst nimmt und Kurvenzüge farbig ausfüllen will, wird schnell ins Fluchen kommen. Man stelle sich bitte einen kleinen Kreis in einem großen Quadrat vor. Nun fülle man den Kreis mit seiner Lieblingsfarbe und das Quadrat mit einer anderen. Nun:

Überall, wo die Füllfarbe auf den Kreis trifft, trifft der TI auf seine Farbgrenzen. Im hochauflösenden Grafikmodus lassen sich pro Character nur noch zwei Farben zuordnen. Da aber jeder Kreis eine schwarze Umgrenzungslinie hat, paßt die zweite Lieblingsfarbe nicht so ganz ins Bild. Das sind dann drei Farben für einen Character, und so etwas sieht der TI nicht gerne. Dann wird er sauer und stellt den betreffenden Character einfach schwarz dar. Also Vorsicht im Umgang mit der Farbe, sonst wird eine elegante Grafik schnell unförmig.

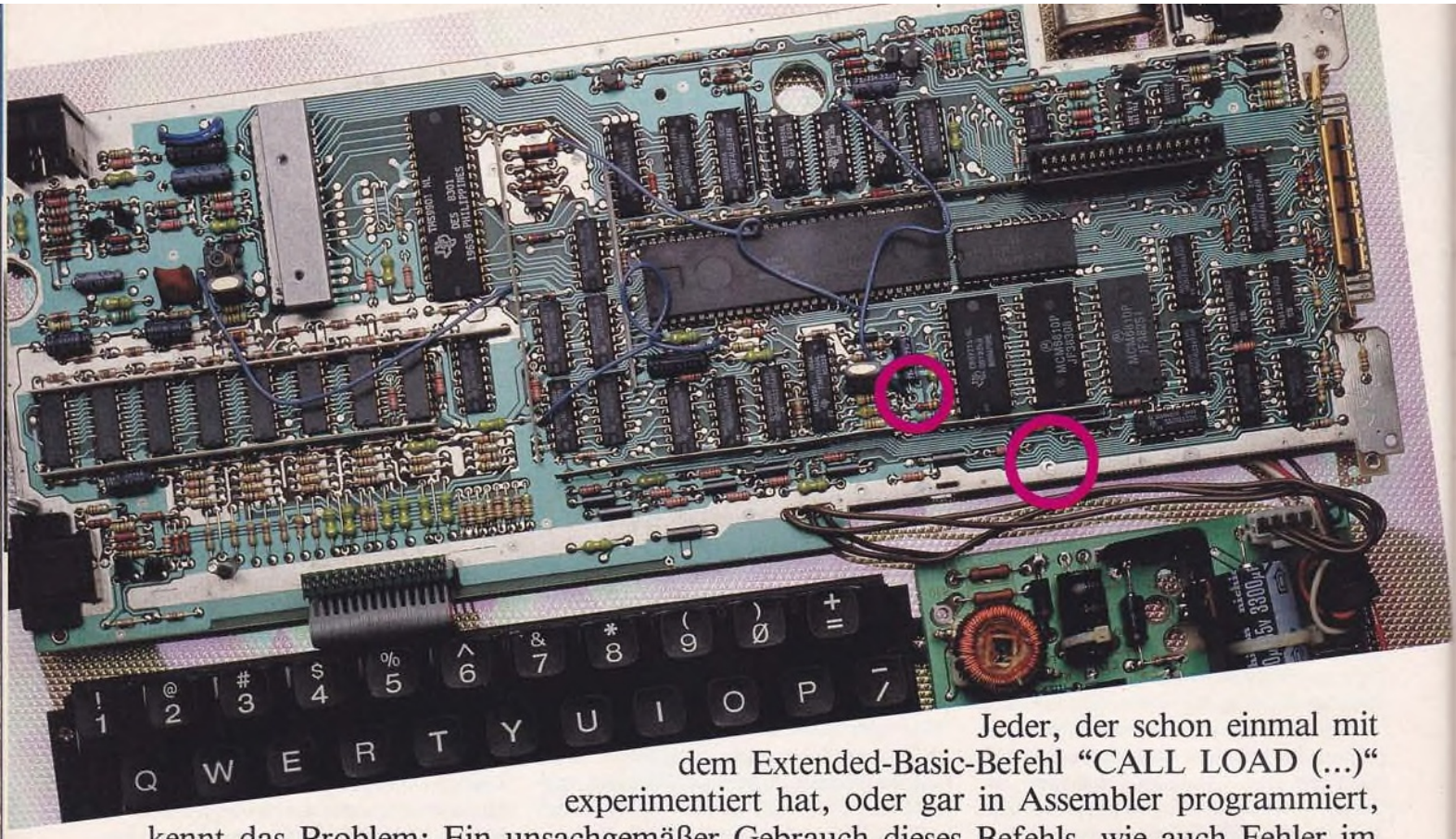
Die Arbeitsgeschwindigkeit läßt keine Wünsche offen, 16 Bit bürgen für entsprechendes Speed.

Am vier Seiten starken Manual ist nichts auszusetzen. Die einzelnen Programmfunktionen werden ausreichend dokumentiert. Aber das Programm ist Dank der guten Menütechnik sowieso selbsterklärend.

Verbindung zum Drucker

Bisher waren die Kullschen Grafikpakete ohne Hardcopymöglichkeit. Der Painter besitzt ebenfalls keine. Wer seinen Drucker unter Dampf halten will, kann sich jetzt eine in Assembler geschriebene Hardcopyroutine zulegen und damit alles auf's Papier bringen, was Exbasic II, 3-D-World, Graphik Master und der Painter hervorzaubern. Zu beziehen ist diese Routine wie auch die genannte Software von Computer Hüsli.

Der Painter wird vom gleichen Haus zusammen mit Exbasic II als Paket für rund 150 Mark angeboten. Der Painter stellt mit Sicherheit eine sinnreiche Erweiterung für den dar, der, ungeübt im Programmieren, lieber Grafiken mit dem Joystick erstellt. Wenn aber die Programmierkünste angewachsen sind, kann der Paketkäufer immer noch auf das Exbasic II zugreifen. Und dann wünschen wir viel Vergnügen. *K.W.*



Jeder, der schon einmal mit dem Extended-Basic-Befehl "CALL LOAD (...)" experimentiert hat, oder gar in Assembler programmiert, kennt das Problem: Ein unsachgemäßer Gebrauch dieses Befehls, wie auch Fehler im Assembler-Programm können den Computer 'sperren'. Unsere Taste bringt einen neuen

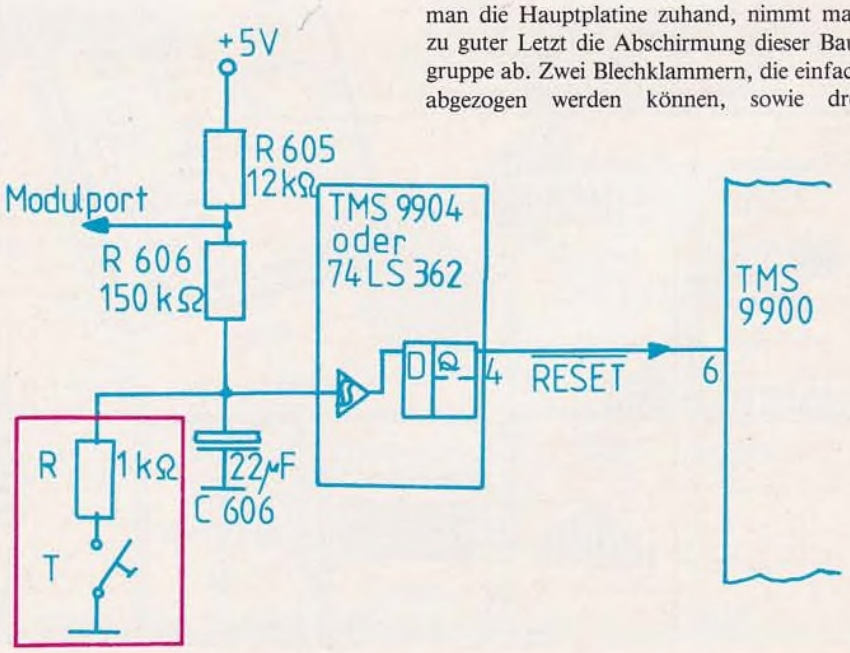
Startschuß zum TI

Ist der Rechner abgestürzt, hilft oft nur noch ganz brutales Aus- und Einschalten der Konsole. Bei einem nicht batteriegepufferten RAM, wie z.B. bei der externen Speichererweiterung, geht so aber das gesamte Programm verloren. Abhilfe schafft hier eine RESET—TASTE. Mit dieser Taste läßt sich das gesamte System in den Anfangszustand, sprich TI-Titelbild, zurücksetzen, ohne die Spannungsversorgung des Speichers abzuschalten. Dadurch bleiben im Maschinencode abgelegte Programme erhalten.

Wie wird nun diese so hilfreiche RESET-TASTE eingebaut?

Um an die beiden Anschlußpunkte (Lötpunkte) zu gelangen, muß der TI 99/4A zerlegt werden. Hierbei gibt es fast keine Probleme. Zuerst werden die sechs Kreuzschlitzschrauben der Bodenplatte entfernt. Diese Platte hebt man ab, indem man vorher den Bedienschieber des Netzschalters nach vorne wegzieht. Zu erkennen sind jetzt das Netzteil, die Tastatur und die im Abschirmgehäuse befindliche Hauptplatine. Das Netzteil ist mit zwei Schrauben befestigt; bei der Tastatur müssen je zwei Schrauben links und rechts entfernt werden, und bei der Hauptplatine sind es drei Schrauben (eine davon in einem Loch der Platine sichtbar). Nun sind noch zwei Steckverbindungen zu trennen. Eine davon verbindet das Netzteil mit der Hauptplatine, die andere die Tastatur mit der Hauptplatine. Das Netzteil und die Tastatur lassen sich somit ohne weiteres entfernen.

Die Hauptplatine kann nur herausgehoben werden, wenn weder ein Modul steckt, noch der Peripheriestecker angeschlossen ist. Hat man die Hauptplatine zuhand, nimmt man zu guter Letzt die Abschirmung dieser Baugruppe ab. Zwei Blechklammern, die einfach abgezogen werden können, sowie drei



Schrauben mit Muttern halten die beiden Bleche zusammen. Zieht man nun noch die Steckleiste des Modulschachtes nach oben heraus, steht dem Einbau der Taste nichts mehr im Wege.

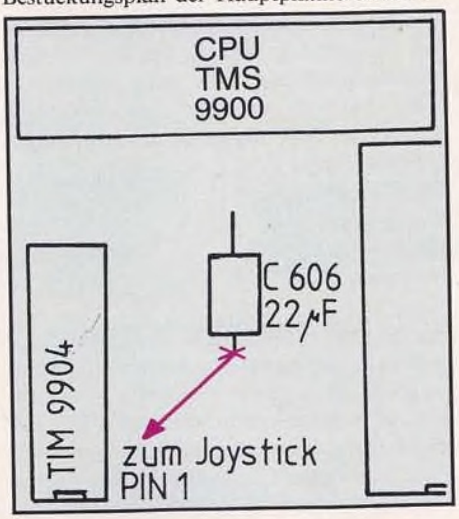
Unsere Schaltskizze auf Seite 90 und einige Erläuterungen dazu sollen die Wirkungsweise der RESET—TASTE erläutern und den Einbau erleichtern.

Der TI 99/4A ist mit einem POWER—ON—RESET ausgestattet. Das heißt, beim Einschalten der Konsole wird der Programmzähler und alle Printer des Computers automatisch auf 'Null' rückgesetzt.

Dies wird mit einer R-C-Kombination erreicht, die an Pin 5 des TIM 9904 angeschlossen ist. Der Kondensator C606 lädt sich nach dem Einschalten über die beiden Widerstände R605 und R606 auf 5V auf. Dabei hängt die Aufladezeit von den Größen dieser drei Bauteile ab. Ab einer bestimmten Spannung legt der nachgeschaltete Schmitt-Trigger seinen Ausgang auf High. Der D-Eingang des Flip-Flop wechselt dabei von Low nach High, wodurch der Q-Ausgang ebenfalls High wird. Der RESET-Impuls ist also somit beendet. Will man nun einen erneuten RESET-Impuls erzeugen, muß nur der Schwellwert des Schmitt-Triggers unterschritten werden, so daß am D-Eingang des Flip-Flop ein erneuter Wechsel stattfindet und der Q-Ausgang für kurze Zeit auf Low schaltet.

Wie im Schaltbild dargestellt, erreicht man dies, indem man den Kondensator über einen Widerstand R und eine Taste T an Masse legt und somit den Kondensator mehr oder weniger entlädt, je nachdem wie lange die Taste gedrückt wird.

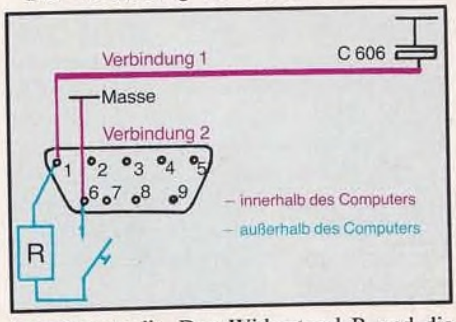
Nach diesen kurzen Erläuterungen nun zum Einbau der Taste. Uns schwebt da eine ganz besonders elegante Lösung vor. Als Hilfestellung soll hier ein Ausschnitt aus dem Bestückungsplan der Hauptplatine dienen:



Um nach außen hin eine saubere Schnittstelle zu erhalten, benutzen wir die zwei freien Anschlüsse der Joystick-Buchse. Es muß also eine Verbindung zwischen C606 und Pin 1 der Joystick-Buchse und zwischen der Masse des Computers und Pin 6 der Joystick-Buchse hergestellt werden. Im Bestückungsplan ist zu erkennen (rotes Kreuz), wo der Verbindungsdraht angelötet werden muß (Lötseite der Platine).

Folgendes Bild verdeutlicht den Verbindungsaufbau:

Die Verbindung zur Außenwelt ist also



jetzt hergestellt. Den Widerstand R und die Taste T kann man z.B. auf einen 9-poligen Stecker löten, so daß je nach Bedarf einmal der Joystick und das andere Mal die RESET-TASTE angesteckt werden kann.

Wer noch Fragen hat, rufe uns an.

Siegfried Fränkl

Achtung!!!

RADIX

RADIX
Bürotechnik
Handelsgesellschaft mbH
Bornstraße 4 · 2000 Hamburg 13
Tel. 040/44 16 95 · Telex 21 36 82 radix d
tägl. 10.00–12.30 + 13.30–18.30 Uhr
Sa. 10.00–13.00 Uhr

Original TI-Produkte · Fremdanbieter für TI 99/4A · Importartikel für TI 99/4A

Umfangreiches Angebot für Texas Instruments

TI 99/4A

Hardware:

TI-Box, Disklaufw., Contr. 32 K-Erw.+10 Disk.	1698,–
32 K-Erw. extern	428,–
Druckerinterface extern	348,–
Drucker GP 50 + Kabel + Druckerinterface extern	798,–
dito mit GP 550	1198,–
dito mit EPSON RX 80	1448,–
dito mit EPSON FX 80	1990,–
Graphik Tableau	248,–
Sprachsteuereinheit	348,–
joyst. zweifach	75,–

Mini Memory **295,–**
Terminal Emulator **99,–**

★★ NEU! ★★

128 K – Karte intern	998,–
USCD-PASCAL 4.0 P-Code Karte, Comp-, Linker Editor	1398,–
Trackball für **TI 99**	225,–
Basic Compiler Disk.	248,–
Basic Combiler für M M.	98,–
Assembler Lernkurs dt. von **IDA Diskette** oder Cas.	98,–

Software:

Editor Assembler	189,–
Extendet Basic orig.	295,–
orig. dito dt. Nachb.	248,–
Parsec	79,–
Alpiner	79,–
Tunnels of Doom	79,–
Adventure Modul	79,–
TI-Logo II dt.	320,–
TI-Writer	320,–
Mathematik I	99,–
Car Wars	39,–
Munch Man	79,–
Tomstone City	39,–

US-Spiele:
Moon Mine, Sewermania, Bigfoot, Meteor Belt, M*A*S*H, Microsurgion, Demon Attack, Hopper, Star Treck, Jawbreaker II, Slymoids, Munchmobile, Moonsweeper, Baseball*, Terry Turtle's Adventure*, Tathom je Modul **49,–**
*** Sprachsteuereinheit notwendig!!!**

Neu! **Atari-Spiele TI 99:**
Assembler – Hardware Zusatz für TI 99 (nur Extendet-Basic notwendig) **149,–**
Defender, Donkey Kong, DIG-DUG, Moon Patrol, Pac Man, Pool Position Jungle Hunt je Modul **99,–**

Sonstiges:
Flugsimulation **49,–**
Staubschutzhaube aus Kunstleder **29,–**
und, und, und, und!

Graphik Tableau für
TI 99 = **248,–**
für VC 64 = **248,–**

SHARP MZ-700
MZ 731, Personal-Computer, 64 kB-RAM inkl. Kassettenrekorder, Farbgrafik-Drucker, eingebauter HF-Modulator und RGB-Anschluß **1190,–**
SFD 700 – Floppy für MZ-700 mit 280 KB **1390,–**

Brother, Silver Reed, Seikosha, HP, 3 M Scotch, Apple, Sanyo, Commodore, Atari

Zubehör:
Monitore, Disketten, Druckerbuffer, Disketten-Organisation, Computer-Möbel, Papier

Abdeckhauben
Kunstleder mit Leinen
für TI 99/4A, CBM 64, Atari 600, MZ 700 **29,90**
FX 80 **39,90** QX 10 **99,00**
Sonderanfertigungen kein Problem!

Preise Stand 1.11.84 Alle Preise incl. MwSt. · Preisliste anfordern! · Lieferung erfolgt per NN oder gegen Verrechnungsscheck.
Bestellungen über DM 500,– werden frei Haus geliefert, unter DM 500,– werden DM 5,– Versandpauschale berechnet.
RADIX Bürotechnik · Bornstraße 4 + Heinrich-Barth-Straße 13 · 2000 Hamburg 13 · Telefon 040/441695 · Telex 213682 RADIX

Händler-Anfragen erwünscht

Mitgedacht

Ich bin Schüler, 13 Jahre alt, und lese seit August begeistert Ihre Zeitschrift. Das Background-Tune Programm aus 11/84 hat mir besonders gut gefallen. Da man bisher nur das ganze Musikstück löschen konnte, habe ich Zeilen dazugeschrieben, mit denen man mit dem Feuerknopf des zweiten Joysticks den zuletzt eingegebenen Ton löschen kann. Hier sind die Programmzeilen, die zusätzlich eingefügt werden müssen:

```
455 IF STRIG(1)=0 AND
    MENGE >0 THEN
    MENGE=MENGE-1:
    POKE 1583+MENGE,0:
    FOR I=1 TO 10:
    SOUND 0,243,10,10:
    NEXT I
456 SOUND 0,0,0,0
```

Da mir 200 Töne zuwenig waren, versuchte ich, das Sound-Maschinenprogramm in einen anderen Speicherbereich zu legen. Leider gelang es mir nicht. Könnten Sie mir bitte schreiben, wie ich die ML-Routine umschreiben muß, daß sie nicht auf Page 6 (1536) sondern auf Page 100 (25600) läuft?
Frank Sambeth, Kirchzarten

Diese Änderungen sind wirklich eine Hilfe bei 'Tippfehlern', jedoch nur, wenn diese Fehler sofort bemerkt werden. Die Routine liegt auf der Seite 6, damit sie wirklich in jedem BASIC-Programm ohne Anpassungen läuft. Die Page 100 wird von einem längeren Programm überschrieben — BACKGROUND-TUNE würde dann einen System-Crash verursachen...
Übrigens: Zwar versuchen wir, CP/T-Programme 'perfekt' zu bringen, doch sind wir für Leserideen, die auch spezielle Bedürfnisse abdecken, stets sehr dankbar. CP/T

Peek und Poke am TI

Vielen Dank für den großartigen Artikel. Kann der Autor mir die verwendete Literatur nennen? Ich vermisse übrigens Artikel zur Assemblerprogrammierung. Ansonsten ist Ihr Heft lobenswert.
Joachim Schulz, Lahr

Hallo Joachim,
Texas Instruments hat nie weitergehende Informationen veröffentlicht, als sie in dem Handbuch zum Editor/Assembler Modul und in den "TI 99/4A console and peripheral expansion system technical data" enthalten sind. Weiter hat Texas Instruments alles mögliche unternommen, um den Blick hinter die Kulissen beim TI 99/4A zu erschweren. So hat es eine ganze Weile in Anspruch genommen, bis die Geheimnisse entschleiert werden konnten. In den nächsten Ausgaben werden wir uns immer tiefer mit dem Innenleben des TI 99/4A beschäftigen. Dabei wird natürlich auch die Assemblerprogrammierung nicht zu kurz kommen.Heiner Martin

Linkshänder-Joystick

Der in der November-Ausgabe vorgestellte Joystick erfüllt zwar seine Aufgabe, wer aber einen Atari-Joystick besitzt, kann es auch ohne Kosten schaffen. Der Stick wird einfach an der Unterseite geöffnet, und nun sind nur noch vier der sechs farbigen Kabel in den Klemmen zu vertauschen: Das blaue Kabel kommt an die Stelle des braunen, das braune an die des weißen, das weiße an die des grünen und das grüne nimmt schließlich den Platz ein, an dem zuerst das blaue saß.
Herman Boeken, Straelen,
Christian Jahns, Salzgitter
Andreas Köhler, Erlangen

Wir freuen uns, daß gleich drei CP/T-Leser unabhängig voneinander uns den gleichen Tip geben. Dufte! Wir meinen aber auch, das unsere Version des Adapters ihre Vorteile hat. Schließlich bekommt ein Linkshänder auch mal von einem Rechtshänder Besuch, oder er will einen neuen Joystick im Laden austesten. Trotzdem. Jetzt kann ja jeder Leser selbst entscheiden, was für ihn die beste Lösung ist. CP/T

Aus für die Hitparade?

Was ist denn eigentlich los mit der TM-Hitparade? Die Leser haben sich doch an die Ausgaben von TM bzw. CP gewöhnt, warum werden denn auf einmal einfach manche Sachen weggelassen? Ansonsten ist die neue Ausgabe ja ganz gut geworden, besonders farbiger.
Thomas Joswig, Nürnberg

Anfragen in Bezug auf die Hitparade gab's in den letzten Wochen reichlich. Ab der nächsten Ausgabe werden wir diese in bewährter Form auch wieder veröffentlichen. Vorausgesetzt, daß wir genügend Einsendungen mit den Titeln Ihrer Favoriten in der Post finden. CP/T

Das ist Ihre Seite, liebe Leser, Forum der Diskussionen. Platz für Fragen und Antworten, Podium für Kritik, Zustimmung, Vorschläge, Anmerkungen und Anregungen. Und sicher auch Ausgangspunkt für den Dialog zwischen Computerspielfreunden. Brieflich oder gar persönlich. Auch bei CP/T gilt: Je kürzer die Zuschrift, desto einfacher haben wir's mit der Veröffentlichung. Und wie üblich müssen wir uns vorbehalten, Briefe gekürzt oder auszugsweise wiederzugeben. Also: Ab geht die Post

an CP/T
Redaktion
Paulstraße 3
2000 Hamburg 1

MUSIPACK V-20 / V-64

MACHEN SIE AUS IHREM COMMODORE VC-20 ODER C-64 EINEN ACHTSTIMMIGEN SYNTHESIZER !!!!

- Keyboard: 49 Tasten
- 8 Synthesizer Presets
- 8 Oszillator-Kombinationen
- 8 Hüllkurven-Varianten
- 8 Spezialeffekte wie Percussion, Vibrato, ändern der Filterfrequenz, etc.
- integrierter Sequenzer (3- bzw. 6 Linien)
- komplett mit Anschlußkabeln und Zubehör

Prospekte gratis von:

wilga profi-shop

4953 Döhren a. d. Weser · Tel. 05705/343

NEWS

Futter für die Turtle

Die amerikanische Originalausgabe, geschrieben von Daniel Watt, wurde in den USA als "Buch des Jahres 1983" ausgezeichnet. Druckfrisch ist die deutsche Übersetzung, die der te-wi Verlag unter dem Titel "LOGO, Computersprache für Eltern und Kinder" jetzt herausgegeben hat.

Auf knapp 400 Seiten werden die einzelnen Programmstufen erklärt, versehen mit Hinweisen auf mögliche Fehlerquellen, Programmiertips sowie Hilfestellungen für Pädagogen beim LOGO-Unterricht.

Die Programme sind in der Commodore-Version geschrieben, jedoch wurden alle Befehlsabweichungen zum Apple-, Atari- und IBM-LOGO direkt neben dem jeweiligen Listing aufgeführt. Die Besonderheiten des TI-LOGO sind im Anhang zu finden.

Das erste der vierzehn Buchkapitel beginnt mit einer allgemeinen Übersicht und der Erklärung und Handhabung der JKP Diskette, was für "Jeder kann programmieren" steht. Auf besagter Diskette soll der Anwender laut Anweisung vorgegebene Programme abspeichern, die in späteren Kapiteln wieder benutzt und als Unterprogramm eingesetzt werden.

Anschließend erfolgen die ersten Turtle-Anweisungen. Zur besseren Übersicht steht zu Beginn jedes Kapitels eine Auflistung der auf den folgenden Seiten erläuterten Befehle.

Alle Anweisungen in diesem Buch werden anhand von Programmierbeispielen erklärt, beziehen sich jedoch hauptsächlich auf die Turtle-Grafik und die damit verbundenen Programmiermöglichkeiten. Im neunten Kapitel sind dann spezifische Bedingungen wie FIRST, LAST, BUTFIRST und BUTLAST in Verbindung mit kleinen Mathe-

matikprogrammen aufgeführt.

Natürlich kommen auch die Spiele nicht zu kurz. Im Kapitel 12 wird demonstriert, wie die Paddle-Abfrage funktioniert und in Prozeduren integriert werden kann. Der letzte Abschnitt schließlich befaßt sich mit der Druckeransteuerung und Listingausgabe. Diejenigen, die mit LOGO ernsthafte Anwendungen, Listenverarbeitung und komplexe Programme erstellen wollen, müssen sicherlich auf ein anderes Buch zurückgreifen. Als Lehrmittel für Kinder dagegen, die sich gerade erst mit dem Computer befassen, bietet es eine wertvolle Hilfe.

Als Hinweis für alle LOGO-Freaks: In den nächsten Ausgaben werden wir den LOGO-Programmierkurs fortsetzen. Wie wär's, wenn Sie uns Ihre selbstgeschriebenen Programme zuschickten?

Neuer Günstig-Knüppel

Er liegt gut in der Hand und hat vier Feuerknöpfe: der neue Joystick von Ascom im Vertrieb von Dynamics, Postfach 112005, 2000 Hamburg 11. Seine vier Saugfüße halten den Stick am Tisch so fest, daß man nach beendetem Spiel die verbliebene Kraft braucht, um ihn wieder abzunehmen. Dabei sollte man den Griff nicht als 'Abreiß-Hilfe' benutzen, denn die Kontakte im Innern sind die gleichen, wie beim guten alten Atari-Joystick und reagieren somit etwas empfindlich.

Nach dem Aufschrauben fielen uns sofort die unbenutzten Leiterbahnen und Bohrungen auf: Alles vorbereitet für das automatische Schnellfeuer! Doch schießen kann man mit diesem Monster auch von Hand schnell genug. Die Feuerknöpfe sind so angeordnet, daß erst nach sehr langem Spiel lahme Finger zu erwarten sind. Der griffige Hebel verleitet zu einem schnellen Spiel, doch bei harten Bewegungen ist die schräge Steuerung etwas ungenau. Kleine, sanfte Bewegungen reichen vollkommen aus. An den beiden Seiten des Gehäuses sind Griffstücke angegossen, mit denen der Stick auch im 'frei Hand'-Betrieb gut zu steuern ist, selbstverständlich auch für Linkshänder.

Aus eigener Erfahrung wissen wir, was mit Joysticks so alles passieren kann (bei uns fliegen die Dinger auch manchmal quer durchs Zimmer). Das Plastik-Material ist dafür allerdings etwas zu hart und dürfte einen Absturz auf den Steinfußboden nicht überleben.

Für den Preis von 26 Mark bekommt man einen richtigen Baller-Freuden-Knüppel, mit dem so manches Spiel regelrecht zum schweißtreibenden Abenteuer wird.

Texas-Tip

In der batteriegepufferten Speichererweiterung der Firma Atronic bleibt nach dem Ausschalten des TI99 ein EX-BASIC-Programm vollständig erhalten. Nach dem Neuanschalten der Konsole werden dann alle Zeiger, die auf dieses Programm weisen, neu gesetzt. Das hat zur Folge, daß der TI das

Programm in der Speichererweiterung nicht mehr sieht. Wie Atronic mitteilte, kann dem jetzt abgeholfen werden.

Bevor die Konsole abgeschaltet, bzw. das EX-BASIC Modul durch den Befehl BYE oder die QUIT Taste verlassen wird, ist folgendes durchzuführen:

```
CALL INIT
CALL PEEK(-31952,A,B,C,D)
::PRINT A;B;C;D
```

Die auf dem Bildschirm ausgegebenen Werte für die Variablen müssen Sie notieren; z.B.: 255,125,255,160

Jetzt können Sie Ihre Konsole getrost abschalten. Nach dem Wiederanschalten der Konsole wird das EX-BASIC Modul angewählt und sofort danach

```
CALL LOAD (-31952,255,125,
255,160)::
CALL LOAD(-25,0)
```

eingegeben. Im ersten CALL LOAD Befehl müssen natürlich die vorher ermittelten Werte eingesetzt werden. In unserem Beispiel waren das 225,125,255,160.

Wird jetzt ein LIST durchgeführt, dann sehen Sie: Ihr gesamtes Programm ist wieder da.

Atronic weist darauf hin, daß bei einem auf solche Art durchgeführten Programmneustart die erste Programmzeile vom Rechner überschrieben wird. Daher erscheint es zweckmäßig, diese erste Zeile als REM-Zeile zu gestalten. Der Verlust einer solchen Zeile dürfte kaum tragisch sein.

Monitore von boston

Die Münchner Firma boston computer präsentiert auf dem deutschen Markt eine neue Serie von Farbmonitoren. Produziert werden die MC 3700-Geräte vom italienischen Hersteller Cabel Electronic.

Die Monitore, die für einen Preis von 789 Mark erhältlich sind, wurden speziell für die Wiedergabe von PAL oder RGB/TTL Signalen entwickelt und sind sowohl als Datensichtgeräte für Computer wie Atari, Commodore, IBM etc., als auch für Videorecorder geeignet.

Der RGB-Monitor erlaubt eine Darstellung von 430 x 312 Pixels, die Bildfrequenz beträgt 50/60 kHz. Allerdings verfügt nur das PAL-Gerät über einen eingebauten Lautsprecher.

Die Bildschirmdiagonale mißt 37 cm, der Bildwinkel hat eine Neigung von 90 Grad. Der dreh- und kippbare Fuß, mit dem die Monitore ausgeliefert werden, gewährleistet ein optimales Sehvergnügen.

Sinclair will Service verbessern

Sinclair will mehr für seine Kunden tun als bisher. Die Garantie für neue Geräte wird darum von bisher sechs Monaten auf ein volles Jahr angehoben. Jedem Computer wird eine Garantie-Anforderungskarte beigelegt, mit der ein Service-Scheckheft von Sinclair angefordert werden kann.

Die Reparatur von Defekten werden sechzehn, im gesamten Bundesgebiet verteilte Geschäftsstellen übernehmen.

Auch im pädagogischen Bereich will Sinclair einen besonderen Service bieten. Kurse zum Thema Computer im Schuleinsatz sollen regional durchgeführt werden. Für Lehrerfortbildungen können Sinclair-Rechner leihweise angefordert werden. Die Anschrift: Sinclair, Kennwort: Computer-Pädagogik, Hessenring 83, 6380 Bad Homburg, Tel.: 06172-25034. Hier sind weitere Informationen zu erfragen.

Keine Grauimporte
Brandheißes Angebot

Sinclair ZX Spectrum 48K & 8 Spielcassetten & Adressverwaltung. Die Spielkassetten enthalten Spiele wie Schach, Flugsimulator, Backgammon **DM 485,–**

Sinclair ZX Spectrum Plus (der Neue mit fest eingebauter Tastatur) & 3 Anwendungsprogramme. Reichhaltiges Soft- und Hardwareangebot – gut und günstig – für Sinclair, IBM, Commodore, Seikosha **DM 599,–**

Spiele für Spectrum:

Hurg – gestalten Sie Ihr eigenes Adventure ohne zu programmieren	**DM 39,–**
Space Shuttle – Flugsimulator	**DM 27,–**
Manic Miner	**DM 24,–**
The Quill Adventure	**DM 39,–**
Penetrator-Scramblespiel	**DM 25,–**
Froggy – bringen Sie den Frosch sicher über die Straße	**DM 23,–**
Mined Out	**DM 19,–**
Night Fite II – Flugsimulator	**DM 35,–**

Spiele für VC 64:

Hover Bovver (Kas.)	**DM 19,–**
Advance of the Mega Camels (Kas.)	**DM 19,–**
Gridrunner (Kas.)	**DM 17,–**
The Hobbit (Kas.) mit Buch	**DM 59,–**
Hungry Horace (Kas.)	**DM 19,–**
Grandmaster Schach (Kas.)	**DM 59,–**
737 – Flugsimulator (Kas.)	**DM 38,–**
Flightsimulator II (Disk.)	**DM 178,–**
(Kas.)	**DM 148,–**
Astrochase (Disk.)	**DM 98,–**
Castle Wolfenstein (Disk.)	**DM 98,–**
Seikosha-Drucker GP 50S	**DM 359,–**
GP 500A m. Interface	**DM 750,–**

Auf alle Geräte erhalten Sie 6 bzw. 12 Monate Garantie. Fordern Sie unseren großen Katalog an (Schutzgebühr 3,50 DM in Briefmarken). Bestell-Annahme tel. 8.00-22.00 oder schriftlich an

PC Software Versand · Im Buchwald 17 · 7000 Stuttgart 1
Tel.: (0711) 46393-1

TOM, der wagemutige Abenteurer, soll die geheimnisvollen Labyrinthe einer Pyramide nach Schatztruhen durchsuchen. Aber es lauern überall gefährliche Wesen, die ihn attakkieren. Ein brandneues spannendes + unterhaltsames Actionspiel mit wunderbarer Grafik + schöner Begleitmusik. Natürlich zu 100% in Maschinensprache. Joystick erforderlich.

VC 20 (+ 16K), C-64 Kassette **29.-** C-64 Diskette **39.-**

TOM ist nur eines unserer vielen Top-Proramme für VC-20 und C-64. Fordern Sie noch heute gegen 2.- DM ausführliche Prospekte über unser Gesamtangebot an.

NEU! Ab sofort sind unsere Programme im *TURBO-Format* abgespeichert. Ohne zusätzliche Hardware werden die Programme von Kassette **10x**, von Diskette **6x schneller** geladen!

Weitere Super-Spiele für COMMODORE-Computer:

VC 20 — je Kassette **25.-**

Battlefield	Ghost Town (16 K)
Bongo (16 K)	Fire Galaxy (16 K)
High Noon (8 K)	Ice Planet (8K)
Max	Time Raiders (8 K)
Star Defender (16 K)	

C-64

		je Kassette	je Diskette
Bongo	Space Pilot	**29.-**	**39.-**
Galaxy	Zaga		
House of Usher			

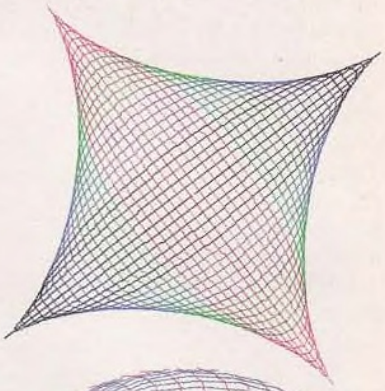
Programmierer gesucht! Händleranfragen erwünscht!

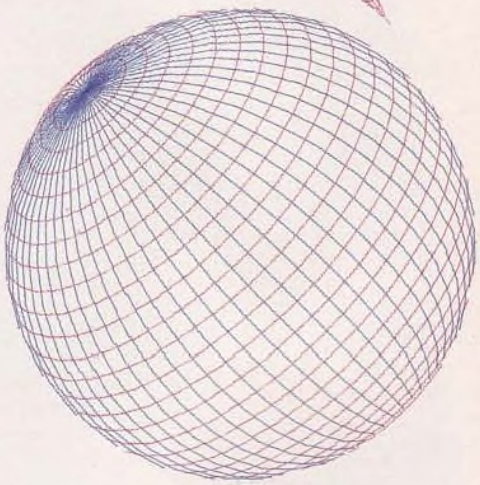
KINGSOFT
»Play it again«
SE · DESIGN

FRITZ SCHÄFER · Schnackebusch 4 · 5106 Roetgen
Telefon 0 24 08/83 19

Klein aber fein!

In unserer Testmangel mußte sich diesmal ein preisgünstiger Printer-Plotter bewähren





Kleine, preisgünstige Plotter mit hohen Leistungen ermöglichen es uns, auch mit dem Heimcomputer farbige Grafiken zu erstellen. Bisher waren wir dabei aber auf recht kleine Formate angewiesen. Nun kommen verstärkt preisgünstige Plotter auf den Markt, die Papiergrößen bis A4 erlauben. Der PP-A4 von Mechatronic ist ein Vertreter dieser Klasse.

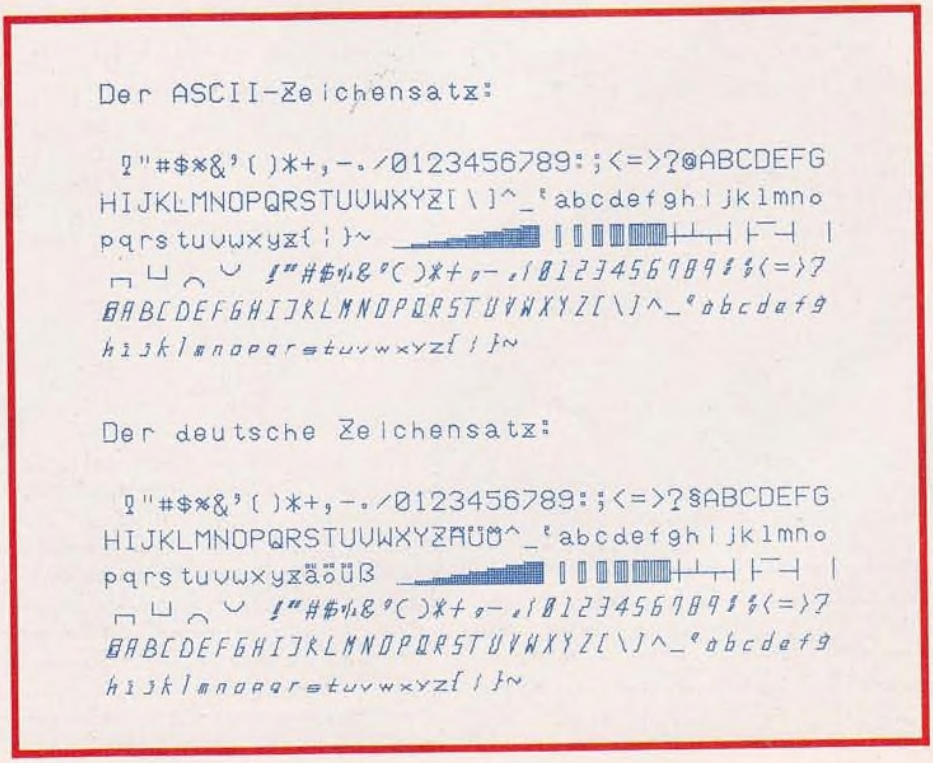
Die erste Überraschung erlebten wir schon beim Auspacken: Aus der Schachtel kam ein handliches Gerät mit einer Grundfläche, die kleiner als eine A4-Seite ist. Auch das Gewicht ist mit ca. 1,2 kg recht niedrig. Neben dem Netzanschluß liegen dem Plotter ein Satz Farbstifte, eine Rolle Papier und die dafür nötige Halterung bei. Bei dem Mustergerät war noch eine englischsprachige Anleitung enthalten, eine Übersetzung ins Deutsche war zum Zeitpunkt unseres Testes leider noch in Vorbereitung.

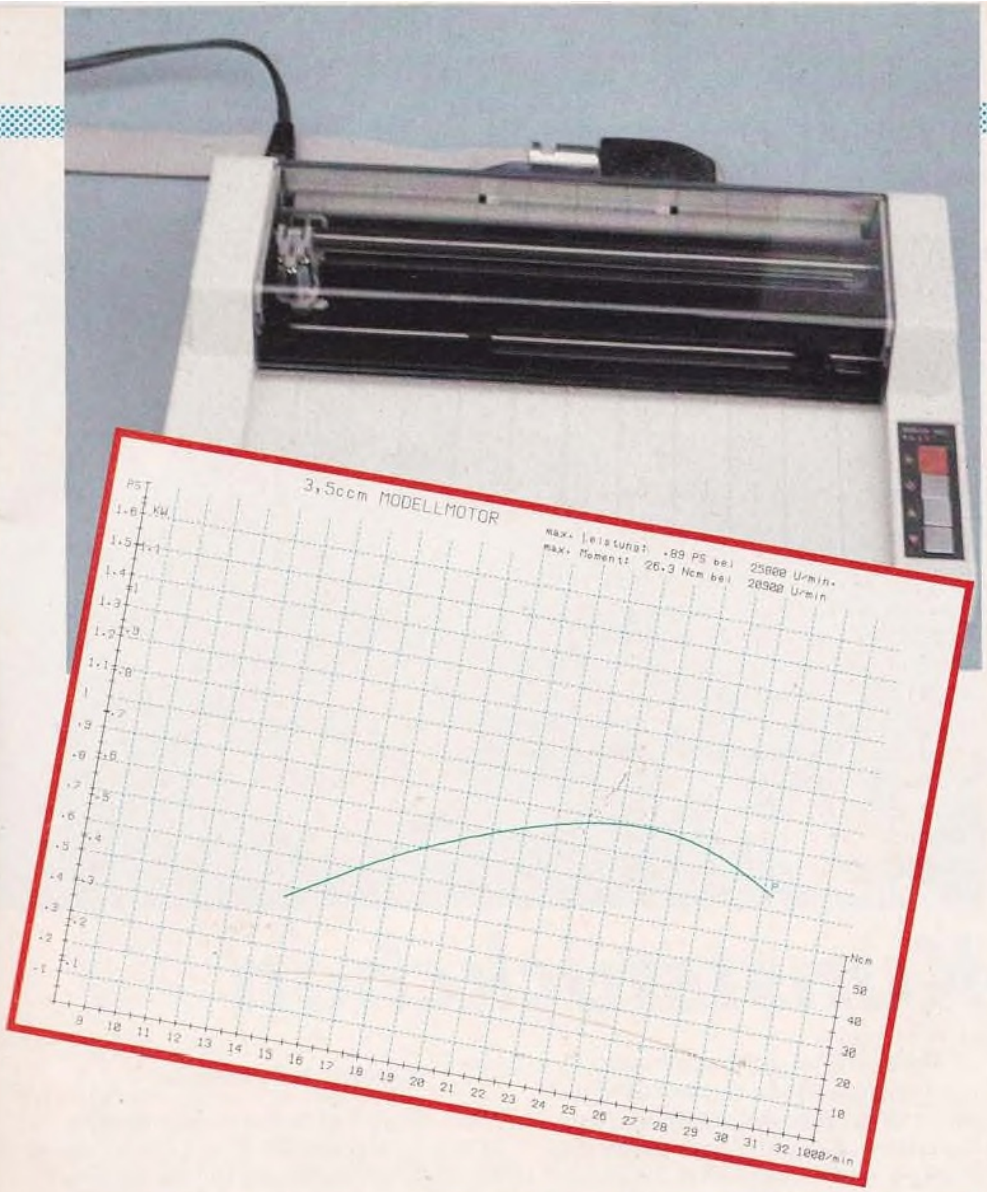
Der PP-A4 arbeitet nach dem Prinzip eines Trommelplotters, das Papier wird also in Längsrichtung bewegt. Die Papierbreite kann zwischen 100 bis 216 mm verwendet werden. Dabei können sowohl Einzelblätter wie Rollenpapier zum Einsatz kommen. Das Zeichnen übernehmen vier kleine Kugelschreiberminen, die im Druckkopf ringförmig angeordnet sind, wie es schon von anderen Plottern her bekannt ist.

Der PP-A4 kann über zwei verschiedene Schnittstellen mit dem Computer verbunden werden: Einer seriellen Schnittstelle mit TTL-Pegel (einen RS232-Adapter gibt es als Zubehör) und einer Centronics-Schnittstelle. Letztere ist sicherlich am weitesten verbreitet und so wurde auch diese zum Anschluß des PP-4A an den Rechner genutzt. Nun ist die PIO des TI 99/4A keine hundertprozentige Centronics-Schnittstelle, und auch die Plotter-Schnittstelle hat ihre Eigentümlichkeiten. So waren unsere ersten Versuche am TI vergeblich. Doch eine kleine Schaltung löste alle Probleme. Das benötigte IC wurde einfach in den Centronics-Stecker eingebaut. Wo beim Anschluß des PP-4A Probleme auftreten, sollten zuerst, wie im Schaltbild gezeigt, die BUSY- und ACKN-Leitungen verknüpft werden. Hilft das nicht, so ist zusätzlich STROBE zu invertieren. Die Versorgung des ICs kann über den an jeder Centronics vorhandenen SV-Anschluß erfolgen.

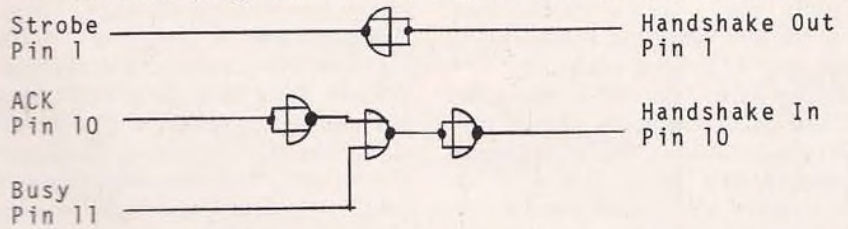
Nachdem diese kleine Hürde überwunden war, konnte der PP-A4 nun auch am TI zeigen, was in ihm steckt. Sein Kürzel steht für "Printer-Plotter A4". Daraus geht schon hervor, daß ein Textmodus vorhanden ist, indem der PP-A4 als Drucker arbeitet. Die Zeile auf einem A4-Blatt umfaßt dabei 80 Zeichen. Der Zeichenvorrat beinhaltet den kompletten ASCII-Satz und zusätzlich für Codes über 128 einige Grafikzeichen und nochmal den ASCII-Satz, diesmal aber in Kursivschrift. Die Kleinbuchstaben in Normalschrift haben Unterlängen. Über DIP-Schalter im Boden des Gehäuses, die von außen zugänglich sind, kann unter vier verschiedenen Zeichensätzen gewählt werden. Zur Verfügung stehen ASCII, Deutsch (mit Umlauten und ß), Englisch und Französisch.

Das Schriftbild der Großbuchstaben kann sich sehen lassen. Die Kleinbuchstaben und die Umlaute in deutschem Zeichensatz sind nicht ganz so schön ausgefallen. Die Kursivschrift ist ziemlich zitterig. Die Steuercodes umfassen Zeilenvorschub, Wagenrücklauf, Rückschritt, Zeilenrückschritt und natürlich





Schaltung für das Kabel:
(Masse und Datenleitungen wie gewöhnlich verbunden,
Stromversorgung des IC von Pin 12 der TI-Schnittstelle)



PP-A4 IC: 74LS02 TI 99/4A PIO

Damit der PP-A4 seine Talente auch am TI unter Beweis stellen kann, sind ein paar einfache Veränderungen am Kabel nötig. Sobald er aber auf Touren gebracht wird, ist mehr aus ihm herausholen, als auf dem ersten Blick in dem kleinen Kasten zu stecken scheint. Die Vignette und die Kugel finden sich als Musterprogramm in der Anleitung und sichern erste Erfolgserlebnisse.

die Umschaltung zwischen Text- und Grafikmodus. Mit Hilfe eines DIP-Schalters oder den Bedienungsschaltern läßt sich die Richtung umstellen, in der das Papier durchläuft. Sehr schnell bringt der PP-A4 seine Schrift nicht zu Papier. Mit ca. vier Zeichen in der Sekunde ist er ganz bestimmt nicht für lange Assembler-Listings geeignet. Allerdings kann er ja nicht nur schreiben.

Im Grafikmodus arbeitet der PP-A4 mit einem Achsenkreuz, welches sowohl in x- als auch in y-Richtung jeweils 1999 Schritte (von -999 bis +999) ermöglicht. Da jeder Schritt 0,2 mm bedeutet, liegt die Höhe des Zeichenfelds bei nicht ganz 400 mm. In x-Richtung ist die ganze Sache aber durch die Mechanik auf 192 mm begrenzt. Dennoch, 192 x 400 mm ist ein ganz schönes Zeichenfeld, größer als ein A4-Blatt, in dem sich auch größere Grafiken gut unterbringen lassen.

Der Befehlssatz im Grafikmodus ist recht umfangreich. Er reicht vom Linienziehen über Farbenwechsel bis hin zu 15 verschiedenen, gestrichelten Linienarten. Text kann sehr einfach zu Papier gebracht werden, dabei ist im Grafikmodus die Schriftgröße in extrem weiten Bereich einstellbar, und die Schrift kann in vier Richtungen geschrieben werden. Mit Hilfe einer Sonderfunktion lassen sich Achsenkreuze sehr einfach zeichnen. Alle Bewegungen können sowohl absolut als auch relativ zur momentanen Position des Druckkopfes ausgeführt werden. Vermißt haben wir allerdings eine Kreisfunktion.

Beachtlich genau

Die Ansteuereung des Plotters geschieht dabei auf eine sehr übersichtliche Art und Weise. Wird z.B. der String "D280,500" an den Plotter ausgegeben, so zeichnet er eine Linie von der momentanen Position des Druckkopfes zum Punkt mit der x-Koordinate 280 und der y-Koordinate 500.

Die Genauigkeit, die der PP-A4 erreicht, ist im Handbuch mit 0,3 mm bei jeglicher Bewegung und mit 0,2 mm Wiederholgenauigkeit angegeben. Diese Werte wurden vom Mustergerät teilweise weit übertroffen. Allerdings muß beim Einlegen des Papiers auf eine gerade Führung geachtet werden, sonst kann es geschehen, daß das Papier während des Zeichnens wegläuft, worunter die Genauigkeit dann sehr leidet. Die Abbildungen zeigen, wie schön der PP-4A zeichnen kann. Bei genauem Hinsehen sieht man allerdings die Schrittgröße von 0,2 mm. Hier ist sicherlich ein Kompromiß geschlossen worden zwischen Preis und aufwendiger Konstruktion. Dies gilt auch für die Zeichengeschwindigkeit von ca. 60 mm pro Sekunde. Für den Hobbybereich ist diese Schnelligkeit aber gut genug. Als Beispiel soll hier das abgebildete Diagramm eines Modellmotors dienen. Im Original ist es genau eine A4-Seite groß und der PP-A4 benötigte etwa zehn Minuten.

Für seinen Preis von knapp unter 900 Mark bietet der PP-A4 von Mechatronic eine ganze Menge. Wenn man keine großen Anforderungen an die Schreibgeschwindigkeit stellt, ist er auch gut als Drucker einzusetzen. Seine Stärken liegen aber ohne Zweifel beim Zeichnen. Die hier erreichte Genauigkeit und Auflösung ist für seinen Preis beachtlich.

Heiner Martin

Turbolader mit Ladehemmung für den TI99

Compiler: Brücke zwischen BASIC und Assembler oder Sturz in den Abgrund?

Wir alle programmieren in BASIC. Nur einige wenige können in Maschinensprache arbeiten. Der TMS 9900 Assembler stellt aber eine Hürde dar, die ohne bestimmte Voraussetzungen nicht übersprungen werden kann.

Editor Assembler Modul, RAM-Erweiterung, Floppy Controller und Floppy und viel Zeit, um die Assemblersprache zu erlernen, sind die Voraussetzungen, um mit seinem TMS 9900 direkt sprechen zu können. Und es ist schon eine arge Plackerei, ein komplexes Spielprogramm in Assembler zu schreiben. Das geht in BASIC wesentlich schneller.

Aber wie übersetzt man ein BASIC-Programm in Assembler? Ganz einfach: Mit einem Compiler.

Wir haben einen Compiler für den TI 99/4A getestet. Zuvor aber möchten wir Ihnen einmal die Aufgaben nennen, die ein Compiler schnell und präzise packen sollte:

- ▷ Compiler laden
- ▷ Programm laden, entweder von Cassette oder Diskette
- ▷ Compilieren und ins RAM laden
- ▷ Testlauf mit Debugger
- ▷ Abspeichern

Einen Compiler, der so funktioniert, würden wir als gut einstufen. Und wenn dann noch Bedienungskomfort vorliegt....das wäre das Tüpfelchen auf dem i.

Dieser Compiler stammt von der SST-Software, Inc. und liegt in Disketten- und Cassettenversion vor. Wir beschäftigen uns nur mit der Diskettenversion, die wahlweise unter dem Mini Memory oder dem Editor Assembler zu betreiben ist. Welche Version Sie bevorzugen, hängt ganz und gar von Ihrer Gerätekonfiguration ab.

Das SST Compiler System besteht aus drei Komponenten:

- ▷ Editor
- ▷ Compiler
- ▷ Loader

Vor den Arbeitsanfang haben die SST-Anbieter das Manual-Studium gesetzt. Da dieses Manual aber in englischer, nicht gerade leichtfüßig zu nennender Sprache geschrieben ist, werden hier schon für manch einen die Schwierigkeiten beginnen.

Woran die Frage zu knüpfen wäre, wann die Softwarehersteller endlich dazu übergehen, ihre Softwarepakete in der jeweiligen Landessprache zu dokumentieren? Dem User würde nämlich so ganz entscheidende Hilfestellung gegeben.

Rein faktisch läßt das System mehr als zu wünschen übrig, wie man unschwer beim Studium des Handbuches feststellt. Wer versucht, sich da einzulesen, wird verstehen, warum wir nicht begeistert waren.

Ein fertiges Basicprogramm ist nicht kompilierbar, jedenfalls nicht in der Form, in der Sie programmiert haben. Sie sollten wissen, daß der Compiler keine Variablendefinitionen mitten im Programm zuläßt. Alle Variablen und Wertzuweisungen müssen am Programmanfang vorgenommen werden.

So einfache Operationen wie A = 10/3 lassen den Compiler böse mit einer Fehlermeldung abbrechen. Obendrein müssen Sie sich entscheiden, ob Ihre Variablen Integer oder Floating Point sein sollen. (Integer ist zwar schneller, aber ungenauer). Damit werden die schönen Variablen zu Konstanten und verlieren ihren Sinn.

Ein einfacher IF THEN ELSE Vergleich treibt den Compiler zum Wahnsinn. Das einzige, was herauskommt, sind Fehlermeldungen. Ein solchermaßen schöner Vergleich ist nur noch gegen 0 oder 1 zulässig. Wer jetzt weiterhin fleißig Variable miteinander vergleicht, wird keine Freude haben.

Es geht leider so weiter. Jeder User weiß die systemeigenen Subroutinen des TI zu schätzen, vereinfachen sie doch mit sehr kraftvollen Befehlen den Programmieralltag.

Dummerweise quittiert der SST Compiler ihre Anwendung nicht mit exzellenter Ausführung, sondern mit Programmabbruch.

Zwar bietet der Compiler Ersatzroutinen an, doch wozu? Ein Compiler sollte doch nur übersetzen und keine neue Sprache schaffen. Auch wenn diese Ersatzroutinen im Handbuch dokumentiert werden, so ist jetzt ersichtlich, was der Compiler vom User verlangt: einen völlig neuen Programmierstil und ein Umschreiben aller Programme, bevor sie compiliert werden können.

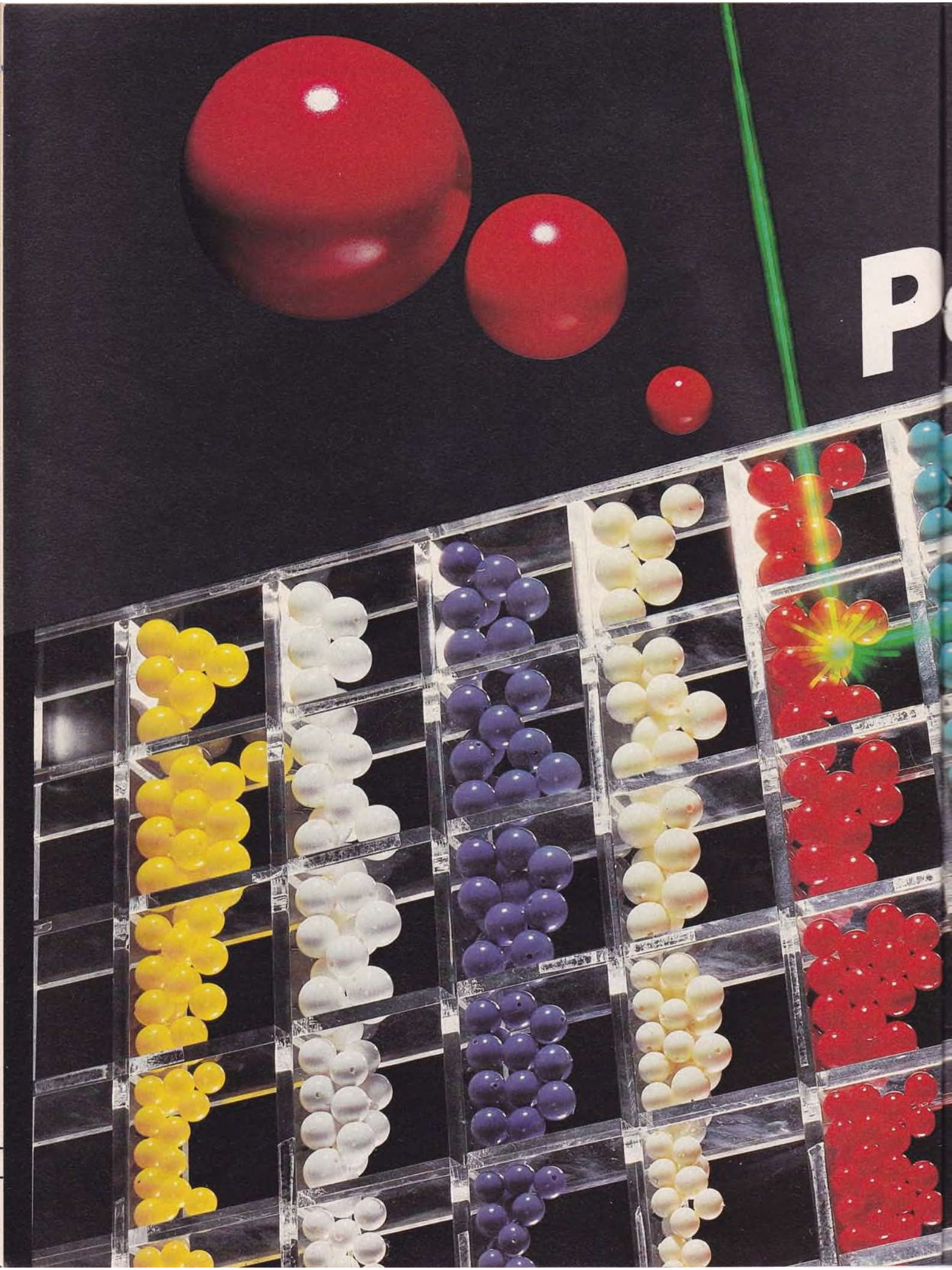
Das Eingeben der BASIC-Programme über den Editor, das anschließende Compilieren und das Laden über den Loader funktioniert zwar einwandfrei, verlangt aber vom User Engelsgeduld. Ein unentdeckter Tippfehler beim Eingeben des Programms oder ein irrtümliches RUN mit nicht resequenztem Editor macht alle Arbeit zunichte. Denn wisse: der Editor darf nur einmal, vorausgesetzt das BASIC-Programm ist fehlerfrei und hat pro Zeile nur ein Statement, resequenzt und mit RUN gestartet werden. Und wenn Sie am Programmende mal wieder das STOP vergessen haben, tja... Pech gehabt, dann wird eben noch Ihr Sofa compiliert. Sie dürfen dann noch einmal anfangen.

Bumm!

Damit platzt der Traum von schnellen Programmen. Jedenfalls solange, bis Sie in schweißtreibender Nachtarbeit Ihre gesamte Programmbibliothek compilierfähig umgeschrieben haben. Das ist ein unmöglicher Aufwand. Ein Compiler, der derartige Eingriffe in die Programmiertechnik verlangt, kann weder als problemlos noch komfortabel eingestuft werden.

Unsere Grundforderung, nämlich ein fertiges BASIC-Programm ohne große Änderungen compilieren zu können, wurde also nicht erfüllt. Fazit: Trotz Niedrigpreis (98 Mark) nicht empfehlenswert.

Klaus Weidemann


P

eek und Poke beim TI 99/4A

Der Wegweiser durch's VDP-RAM in X-BASIC

Seit Produktionseinstellung des TI 99/4A ist jetzt etwa ein Jahr vergangen. Damals sah für die TI 99/4A-User die Zukunft düster aus. Doch genau das Gegenteil ist der Fall: Heute gibt es mehr qualitativ gute Software und mehr Hardware als je zuvor, und das Angebot wird ständig größer. Es hat zwar ziemlich lange gedauert, aber jetzt sind die Fesseln, die Texas Instruments dem 99/4A angelegt hatte, endlich gelöst, die User können sich freuen.

Doch zum Thema: In Computer Praxis 10/84 besprachen wir die Routinen POKEV und PEEKV für Extended Basic und hatten uns abschließend noch kurz mit dem Inhalt des VDP-RAM's unter Extended Basic beschäftigt. Darauf gehen wir heute noch näher ein, wobei wir davon ausgehen, daß die Routinen PEEKV und POKEV geladen sind.

VDP›0000 bis ›02FF (dez. 0 bis 767):
Hier liegt der Bildschirminhalt. Dabei sind die Zeichen nicht im ASCII-Code abgelegt, sondern mit einem Offset von dez. 96 bzw. ›60. Da dieser Offset immer wieder Schwierigkeiten bereitet, hier nochmals ein Beispiel: Um das Wort "COMPUTER" auf den Bildschirm zu poken, müssen wir

CALL LINK("POKEV",355,163,175,173,176,181,180,165,178)

durchführen. Dabei ist 355 die Bildschirmadresse (0 ist die linke, obere Ecke des Bildschirms und 767 die Ecke rechts unten). Der Wert 163 ergibt sich aus dem ASCII-Wert für C von 67 plus dem Offset von 96. 175 setzt sich aus dem ASCII-Wert 79 für das O und wieder dem Offset von 96 zusammen, usw. Das Leerzeichen hat den ASCII-Wert 32; plus dem Offset von 96 ergibt dies 128. Um den Bildschirm zu löschen, können wir also auch folgende Zeile verwenden:

FOR X=0 TO 767 :: CALL LINK("POKEV",X,128) :: NEXT X

Hier ist CALL CLEAR aber sicherlich um einiges sinnvoller.

VDP›0300 bis ›0370 (dez. 768 bis 880)
Diesen Bereich nennt Texas Instruments die Sprite Attribute Table, in dem die Lage, der Charaktercode und die Farbe des Sprites gespeichert sind. Ohne Sprites hat diese Tafel folgendes Aussehen: 192,0,0,0 und das 28 mal. Für jedes Sprite also vier Bytes, die der Reihe nach die Lage in Y-Richtung, X-Richtung, den Charaktercode und die Farbe enthalten. Sprite #1 liegt dabei ab 768, Sprite #2 ab 772 usw. bis Sprite #28 ab 876. Die Y-Lage differiert von den Angaben in Extended Basic. Die Zeile 1 ist beim Poken 255, die Zeile 2 0, die Zeile 3 1, usw. Bei der X-Richtung müssen wir von dem Wert in Extended Basic einfach 1 abziehen, um die richtige Angabe für das Poken zu erhalten. Der Charaktercode enthält den gleichen Offset, wie schon beim Bildschirminhalt beschrieben. Das letzte Byte enthält die Farbe, dabei gilt auch wieder: Basicwert minus 1. Ein

CALL LINK("POKEV,768,100,140,138,6)

ist also gleichzusetzen mit

CALL SPRITE(#1,42,7,102,141)

Wollen wir das Sprite wieder löschen, so lautet der Befehl dazu:

CALL LINK("POKEV",768,192,0,0,0)

Der direkte Zugriff auf diese Speicherstellen gibt uns hier eine Möglichkeit, die Extended Basic nicht bietet. Normalerweise gehen alle bewegten Sprites sauber aus dem rechten Bildschirmrand heraus, springen aber links

Peek und Poke beim TI 99/4A

in das Bild herein. In dem Byte, das die Farbe des Sprite bestimmt, befindet sich noch ein Bit, das early clock bit, welches die ganze Sache umdreht; d.h. wenn dieses Bit gesetzt ist, wird das Sprite am linken Bildschirmrand gleichmäßig hereinlaufen, springt aber dafür am rechten Rand heraus. Der VDP bildet das Sprite dann 32 Spalten nach links versetzt ab. Um dieses zu realisieren, müssen wir bei der Farbe einfach 128 hinzuzählen. Im obigen Beispiel lautet es dann also 134 statt 6.

Auf 880 steht der Wert 208, der uns in den letzten Folgen schon einmal begegnet ist; er verhindert die weitere Abbildung von Sprites und darf nicht verändert werden.

VDP› 0371 bis › 03EF (dez. 881 bis 1007):

Dieser Bereich wird vom Extended Basic zum Zwischenspeichern wichtiger Systeminformationen benutzt. Da diese im wesentlichen nur innerhalb von Basic-Befehlen und bei einer Unterbrechung des Programms genutzt werden, bringen uns diese Speicherstellen momentan noch nicht viel. Auf die eine oder andere werden wir in einer späteren Folge noch zurückkommen.

VDP› 03F0 bis › 077F (dez. 1008 bis 1919)

Hier sind die Definitionen für die Charaktere abgelegt. Beginnend bei 1008 für den Charaktercode 30 (Cursor), der auf im Bildschirmspeicher mit dem Wert 126 beschrieben wird. Die Definition eines Zeichens umfaßt jeweils acht Byte. 126 mal 8 ist ja 1008, wir sehen also, warum der Offset notwendig ist. Die acht Byte der Definition ergeben sich aus dem im Basic verrwendeten CALL CHAR Befehl.

CALL CHAR(32,"FF00FF00FF00FF00")

entsprechen

CALL LINK("POKEV",1024,255,0,255,0,255,0,255,0)

Der String in CALL CHAR entspricht ja hexadezimalen Zahlen, und diese müssen wir zum Poken wieder in dezimale Zahlen umwandeln. Die Anfangsadresse der Definition errechnet sich jeweils aus ASCII-Code mal 8 plus 768.

Mit POKEV haben wir nun auch Zugriff auf den Cursor. Ein holländischer User-Club hat einmal eine nette Veränderung vorgeschlagen, die verblüfft:

CALL LINK("POKEV",1008,60,126,219,255,231,189,195,126)

Übrigens wird die Cursordefinition bei einer Programmunterbrechung nicht auf den Ausgangswert zurückgesetzt.

Ebenfalls einfach wird die Umwandlung einer Dezimalzahl in eine Hexzahl. Folgendes Programm nutzt die Möglichkeit, mit CALL CHARPAT den Definitions-String zu erhalten:

```
100 CALL CLEAR :: INPUT
    "DEZIMAL: ":A
110 IF A < 0 OR A > 65535 THEN 100
120 CALL LINK("POKEV",1912,INT
    (A/256),A-(INT(A/256))
130 CALL CHARPAT(143,B$)::PRINT
    "HEX: ":SEG$(B$,1,4)
140 END
```

Solange keine Spritebewegungen erfolgen, können auch die Charaktersätze 15 und 16 mit den ASCII-Werten 144 bis 159 benutzt werden. Aber die Definitionen liegen im Bereich der Spritebewegungstabellen, so daß vorher unbedingt CALL LOAD(—31878,0), um jede Spritebewegung zu unterbinden, durchgeführt werden muß. Auch müssen, bevor dann wieder Sprites bewegt werden, alle Definitionen gelöscht sein.

VDP› 0780 bis › 07FF (dez. 1920 bis 2047)

Je Sprite sind hier vier Bytes abgelegt, beginnend auf 1920 für Sprite #1, Sprite #2 ab 1924 usw, die die Geschwindigkeit bestimmen. Das erste Byte ist dabei die Geschwindigkeit in vertikaler (Y) und das zweite Byte diejenige horizontaler (X) Richtung. Die anderen beiden Bytes werden nur vom Betriebssystem zum Errechnen der genauen Lage des betreffenden Sprites benutzt und brauchen uns nicht weiter zu interessieren, da die Werte nicht weiter von Bedeutung sind.

Bleiben wir bei dem Beispiel eines Sprites von oben und verleihen ihm Bewegung:

CALL LINK("POKEV",1920,40,80)
CALL LOAD(—31878,1)

Letzterer Befehl ermöglicht dem Texasfreund erst die Spritebewegung.

Wollen wir ihn wieder stoppen:

CALL LINK("POKEV",1920,0,0)
CALL LOAD(—31787,0)

Die Bewegungstabellen für die 28 Sprites des Extended Basic gehen nur bis einschließlich der Adresse 2031, so daß hier noch ein paar Bytes 'frei' sind.

VDP› 0800 bis › 081F (dez. 2048 bis 2079)

Auch bei der Farbtafel sind die ersten fünfzehn Bytes 'frei', da im Extended-Basic ja durch den Offset nur die Charakter-Codes ab 126 Verwendung finden. Die Bestimmung der Charaktersätze ist wie beim CALL COLOR Befehl, d.h. der Cursor und das Randzeichen sind Charaktersatz 0, und dann sind jeweils acht Zeichen zu einem Charaktersatz zusammengefaßt. Auf Adresse 2063 im VDP-RAM steht die Farbinformation für den Charaktersatz 0, wobei für jeden Satz nur ein Byte benötigt wird. Der Wert ergibt sich aus der Vordergrundfarbe minus 1 mal 16 plus der Hintergrundfarbe minus 1. Wollen wir also einen roten Cursor haben, müssen wir ganz schlicht und einfach

CALL LINK("POKEV",2063,112)

eingeben. Der Wert von 112 errechnet sich dabei aus 7 (für Rot) minus 1 mal 16 plus 1 (Hintergrund Transparent) minus 1.

VDP› 0820 bis › 0957 (dez. 2080 bis 2391)

Nun braucht das Extended Basic noch einen größeren Zwischenspeicher für die Eingabe der Programmzeilen und der Verarbeitung dieser zu einem Basicprogramm. Von 2080 bis 2239 befindet sich der Pufferspeicher, der zur Umwandlung der Eingabezeile in die Programmzeile verwendet wird. Von 2240 bis 2391 ist die letzte Eingabe abgespeichert, die mit der Taste REDO(F'ctn 8) wieder auf den Bildschirm geholt werden kann.

VDP› 0958 bis › 3FFF (dez. 2392 bis 16383)

Ab 2392 befindet sich der Value-Stack; das ist ein Stack, der vom Extended Basic intensiv genutzt wird und in dem unter anderem diverse Pointer auf String- und numerische Werte zu finden sind; ferner werden in dem Stack auch die Hilfswerte bei FOR-NEXT Schleifen abgespeichert, und die Rücksprungadressen bei Unterprogrammen sind darin enthalten. Der restliche Bereich des VDP-RAM's wird zur Speicherung der Stringvariablen, der Listen für Variablen, Unterprogramme und Dateien verwendet. Hier befinden sich bei vorhandener Speichererweitung das Programm, die Zeilenliste und die numerischen Variablen.

Zu guter Letzt ist bei angeschlossenem Diskettenlaufwerk noch ein Bereich im VDP-RAM für den Diskettenkontroller reserviert. Die Größe dieses Bereiches ist abhängig davon, ob mit CALL FILES zusätzlicher Platz für Dateien reserviert wurde. Beim Einschalten des Systems wird ja einmal CALL FILES(3) automatisch durchgeführt, dann ist ab › 37D8 (dez. 14296) bis zum Ende des VDP-RAMs der Diskettenkontroller am Werk. Wir sollten, wie schon beim letzten Mal erwähnt, in diesen Bereich nicht hineinpoken, da eine Diskette sonst leicht zerstört ist — und wer hat das schon gern!.

Kehren wir zum Schluß dem VDP-RAM den Rücken zu, und gehen nochmal zum Interrupt Flag Byte auf › 83C2 (dez. —31806) ein. Wir hatten schon einmal die Möglichkeit erwähnt, die Wirkung der Quit-Taste mit

CALL LOAD(—31806,16)

und die Sprite-Bewegung mit

CALL LOAD(—31806,64)

abzuschalten. Nun,

CALL LOAD(—31806,32)

schaltet auch den automatischen Soundablauf ab. Wird danach ein CALL SOUND durchgeführt, tönt es ewig.

Heiner Martin

OBJEKTCODE-EDITOR

TI-Maschinensprache ohne Assemblierung

Objektcode-Editor

```
100 !{23*(*)}
110 !* OBJEKTCODEGENERATOR
120 !*
130 !{23*(*)}
140 !* H.AMSHOVE
150 !* VERS. 1.0  @8.8.1984
160 !{23*(*)}
170 ! EINGABE DES OBJEKTCODE
180 ! LISTINGS OHNE ZEILENNR
190 !
200 ! ENDE  DER EINGABE DURCH
210 ! <ENTER> BEI DER WERT
220 ! ABFRAGE
230 !{23*(*)}
240 CALL CLEAR
250 DISPLAY AT(2,1):"FILENAME" ::
    ACCEPT AT(2,12): FILE$
260 OPEN #1:FILE$,UPDATE,RELATIVE,
    DISPLAY ,FIXED 80
270 DISPLAY AT(4,1)ERASE ALL:"1--EINGABE":
    "2--KORREKTUR":"3--ERGAENZEN":
    "4--LISTEN":"5--ENDE"
280 DISPLAY AT(10,1):"BITTE WAEHLEN" ::
     ACCEPT AT(10,15)SIZE(1)
    VALIDATE("12345"):EIN$
290 ON VAL(EIN$)GOSUB 330,430,570,640,310
300 GOTO 270
310 CLOSE #1
320 END
330 !EINGABE
340 CALL CLEAR :: ZNR=0
350 ZNR=ZNR+1
360 DISPLAY AT(2,1)ERASE ALL:"ZEILE:";ZNR
370 DISPLAY AT(4,1):"WERT:"
380 GOSUB 750
390 IF A$="" THEN RETURN
400 GOSUB 850
410 PRINT #1:DAT$
420 GOTO 350
430 !KORREKTUR
440 DISPLAY AT(2,1)ERASE ALL:
    "ZEILE:" :: ACCEPT AT(2,12):ZNR
450 INPUT #1,REC ZNR-1:DAT$
460 DISPLAY AT(6,1):SEG$(DAT$,1,20)
470 DISPLAY AT(7,1):SEG$(DAT$,21,20)
480 DISPLAY AT(8,1):SEG$(DAT$,41,20)
490 DISPLAY AT(9,1):SEG$(DAT$,41,10)
500 GOSUB 750
510 GOSUB 850
520 PRINT #1,REC ZNR-1:DAT$
530 DISPLAY AT(24,1):"W(EITER E(NDE"
540 ACCEPT AT(24,15)SIZE(1):EIN$
550 IF EIN$="W" THEN 440
560 RETURN
570 !ERGAENZEN
580 ZNR=0
590 IF EOF(1)THEN 630
600 INPUT #1:DAT$
610 ZNR=ZNR+1
620 GOTO 590
630 GOTO 350
640 !AUSLISTEN
650 DISPLAY AT(2,1)ERASE ALL:
    "LISTFILE" :: ACCEPT AT(3,1):LIST$
660 IF LIST$="" THEN 690
670 OPEN #2:LIST$
680 IF EOF(1)THEN 730
690 INPUT #1:DAT$
700 IF LIST$="" THEN PRINT DAT$ ELSE
    PRINT #2:DAT$
710 CALL KEY(0,KEY,ST):: IF ST<>0 THEN 710
720 GOTO 680
730 IF LIST$<>"" THEN CLOSE #2
740 RETURN
750 !EINGABE WERTE
760 ACCEPT AT(6,1)SIZE(-20):A$
770 IF A$="" THEN 810
780 ACCEPT AT(7,1)SIZE(-20):B$
790 ACCEPT AT(8,1)SIZE(-20):C$
800 ACCEPT AT(9,1)SIZE(-9):D$
810 DISPLAY AT(24,1):"K(ORR W(EITER"
820 ACCEPT AT(24,7)SIZE(-1):EIN$
830 IF EIN$="K" THEN 760
840 RETURN
850 !DATENSATZ ERSTELLEN
860 DAT$=A$&B$&C$&D$
870 DAT$=DAT$&RPT$(" ",76-LEN(DAT$))
880 Z$=STR$(ZNR)
890 Z$=RPT$("0",4-LEN(Z$))&Z$
900 DAT$=DAT$&Z$
910 RETURN
```

Das Programm Objektcode ermöglicht die Eingabe eines Objektcodelistings, wie es normalerweise vom Editor/Assembler erzeugt wird. Es erstellt ein File, das anschließend vom ExBASIC-Loader genauso geladen werden kann, wie das Objektcodefile des Editor/Assemblers. Der Vorteil gegenüber einer POKE-Liste liegt darin, daß verschiedene auf diese Weise eingegebene Programme aneinandergehängt werden können. Desweiteren werden hier Eingabefehler durch den Loader erkannt und angezeigt, so daß Fehlersuche einfacher ist.

Das Programm ermöglicht:

▶ Eingabe eines Files: Eingegeben wird der Objektcode. Am einfachsten erfolgt das, indem man das Listing in Spalten zu 20 Character unterteilt, da die Bildschirmeingabe eine Objektcodezeile in vier Eingabezeilen aufteilt, um Eingabefehler möglichst früh zu erkennen. Die Zeilennummer braucht nicht miteingegeben zu werden, da sie automatisch erzeugt wird.

▶ Korrektur eines Files: Sollte sich ein Fehler eingeschlichen haben, kann das File nachträglich editiert werden. Sie geben die Zeilennnummer an, der Inhalt der Zeile wird angezeigt. Die Bildschirmzeilen, die nicht geändert werden sollen, brauchen Sie nicht neu einzugeben; stattdessen einfach ‹ENTER› drücken.

▶ Ergänzen eines Files: Sollten Sie bei der Eingabe gestört werden, so können Sie die Eingabe beenden und zu einem anderen Zeitpunkt fortsetzen.

▶ Listen eines Files: Unser Programm ermöglicht das Auflisten des Files, um eine bessere Kontrolle zu erhalten.

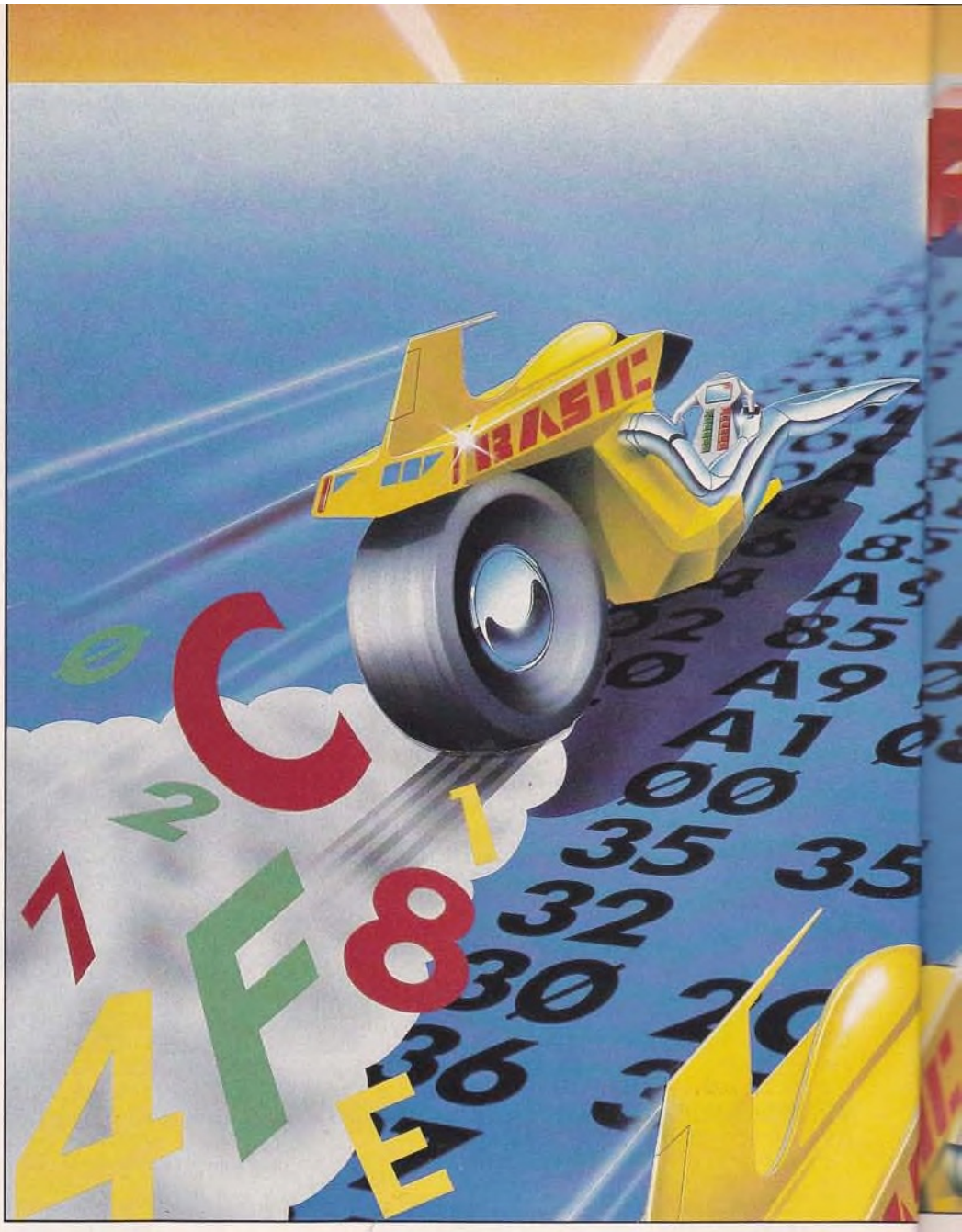
Damit Sie unseren Objektcode-Editor auch gleich in der Praxis ausprobieren können, geben wir Ihnen ein Beispiel-Codelisting. 'Block-clear' hilft bei der Erstellung einer Bildschirmmaske. Sie kann auf den Stand zurückgebracht werden, auf dem sie vor der Eingabe war. Durch Eingabe von Anfangs- und End-Zeile/Spalte wird bestimmt, welches Feld auf dem Bildschirm gelöscht wird.

Block-clear wird mit CALL LINK ("CLRB", row1, row2, col1, col2) aufgerufen, wobei row1 = Anfangszeile, row2 = Endzeile, col1 = Anfangsspalte und col2 = Endspalte bedeutet.

Durch unseren Objektcode-Editor kann das Programm auch ohne Editor/Assembler geladen werden. Weitere Anwendungsbeispiele folgen. *H. Amshove*

Anwendung Blockclear

```
00118BLOCK    A0000A0
020A0022A0024A0026A0
028B8000B0300B0000B0
4207F2C7F        0001
A0030B200CB0420B2018
B12B8B0300B0002BC4A0
B834AB8ED2B1104B0200
7F2E4F           0002
A0046B1C00B0420B2034
B8ED2B1504B0200B1C00
B0420B2034B045BB02E0
7F2F1F           0003
A005CC0000B0200B0000
B0201B0001B0202C0020
B06A0C002AB0019B0000
7F357F           0004
A0072B0620C0020B0201
B0002B0202C0022B06A0
C002AB0019B0000B0620
7F353F           0005
A0088C0022B0201B0003
B0202C0024B06A0C002A
B0021B0000B0620C0024
7F34FF           0006
A009EB0201B0004B0202
C0026B06A0C002AB0021
B0000B0620C0026B8820
7F32FF           0007
A00B4C0022C0020B1504
B0200B1F00B0420B2034
B8820C0026C0024B1504
7F336F           0008
A00CAB0200B1F00B0420
B2034BC120C0022B6120
C0020B0584BC160C0026
7F30BF           0009
A00E0B6160C0024B0585
BC1A0C0020B0A56BA1A0
C0024BC1C5BC060C0028
7F2B9F           0010
A00F6BC006B0420B2020
B0586B0607B16FAB05A0
C0020B0604B16EEB04C0
7F2BAF           0011
A010CBD800B837CB02E0
B83E0B0460B00707F81F
F
5005ACLRB  7FD5BF
:          99/4 AS
```



Sprechen Sie

Der TI spricht mit vielen Zungen. Nun hat er noch eine dazugelernt: FORTH!!!

Charles Moore begann 1969 eine Computersprache zu entwickeln, die schnell, leicht zu erlernen und außerdem für die Steuerung von Radio-Teleskopen geeignet sein sollte. Keine leichte Arbeit, denn damals waren die Rechner wesentlich größer als heute und auch sehr viel schlechter dokumentiert.

Trotzdem war er erfolgreich; er taufte das Resultat seiner Anstrengungen FORTH. Oftmals wird FORTH als die Computersprache der vierten Generation betitelt, als die Sprache der Zukunft. Seit FORTH auf dem Markt ist, hat diese Sprache einige Entwicklungsstufen durchgemacht und ist mittlerweile aus den Elfenbeintürmen der Wissenschaftler auch zu uns Normal-Usern herabgestiegen. Doch benutzen es viel zu wenige.

Aus englischen Landen frisch auf den deutschen Tisch kam denn auch der erste Homecomputer, der diese sagenumwobene Sprache fest implementiert hatte. Leider war diesem Rechner wenig Erfolg beschieden. Es hatte sich allerdings schon früh eine Gruppe an FORTH interessierter Leute zusammengefunden, die um die Fähigkeiten dieser Sprache wissend, sich um die Weiterentwicklung und auch Standardisierung ihres Zöglings kümmerten. Aus dieser Arbeit entstand das fig-FORTH. ('fig' heißt zu Neudeutsch 'forth interest group').

Die Arbeit dieser Gruppe blieb nicht ohne Folgen. Aus dem fig-FORTH wurde ein FORTH-79, ein FORTH-83 — die neueste Version übrigens — und irgendwann wachten die Homecomputerhersteller aus ihren Basic-Über-Alles-Träumen auf. Mittlerweile gibt es für alle gängigen Prozessoren FORTH. Angeboten wird es als Assemblerlisting zum Selbst-Installieren, auf Diskette oder auch als Modul. Für den TI User sind alle drei Wege offen. Der engagierte Hacker findet seine Freude am Assemblerlisting und der Programmierer an der Disketten- oder Modulversion. Doch was ist eigentlich so Besonderes an diesem FORTH?

Das Besondere

FORTH ist im Gegensatz zu BASIC eine hochstrukturierte Sprache, die einen unglaublichen Geschwindigkeitsgewinn bringt. FORTH ist ca. 40 — 50 mal schneller als das TI-BASIC. Damit wird auch schon deutlich wo FORTH anzusiedeln ist. Nämlich in unmittelbarer Nähe zum TMS 9900-Assembler. Wobei aber wiederum dem Herrn Moore zu danken ist, denn Forth lernt sich leichter und schneller als Assembler. Aber ganz ohne Asemblerkenntnisse kommt der Anwender nicht aus. Zumal das uns vorliegende TEX-FORTH (vom Computer Hüsli) reichlich davon Gebrauch macht. Eine intime Kenntnis der Systemadressen ist unbedingt notwendig, da FORTH ständig auf diese Adressen zurückgreift. Zum anderen verlangt FORTH ein ganz neues Programmiergefühl vom Anwender. FORTH arbeitet mit zwei Stacks, dem Data und dem Returnstack. Diese beiden Stapel sind das Kernstück. Auf den Stack werden alle notwendigen Datas und Adressen gelegt, die der Rechner zur Abarbeitung eines Programms benötigt. Wie das genau geschieht, kann aus der weiter unten angegebenen Literatur zum Thema FORTH entnommen werden.

FORTH verzichtet total auf das uns so gewohnte GOTO Statement. Dafür ist der Programmierer in der Lage, sich Prozeduren zu bauen, die wiederum andere aufrufen. So kann man z.B. genau die Prozedur aufrufen, von der der Aufruf erfolgt. Das nennt man dann rekursives Programmieren. Ein kleines Beispielprogramm soll Ihnen diese Rekursivität erläutern.

```
: FACT DUP 1 > — IF DUP 1 — MYSELF
* ENDIF ;
```

Das obige Programm berechnet Fakultäten. Mit dem Wort 'Myself' wird das Programm von sich selbst erneut aufgerufen, bis

e Forth?

das Aussteigekriterium erreicht ist. Programmieren Sie soetwas mal in Basic. Das einfache Fakultätenprogramm zeigt sehr deutlich die Leistungsfähigkeit von FORTH. Es ist aber noch anderes an diesem Beispiel zu erkennen, nämlich daß eine ganze Reihe neuer Wörter erlernt werden muß. Befehlswörter, die ein fester Bestandteil der jeweiligen FORTH-Version sind. Darüber hinaus lassen sich jederzeit neue Wörter definieren, die dann, geschickt miteinander verkettet, ganze Programme bilden.

Sie ahnen schon, was da auf Sie zukommt? Das wilde, zügellose Drauflosprogrammieren, wie es beim BASIC möglich ist, ist bei FORTH nicht drin. Nur exaktes Planen und Entwerfen eines Programms, unterstützt von Flußdiagrammen führt zum gewünschten Ziel. Zum schließlich lauffähigen Programm.

Programmieren mit Hindernissen

Umgekehrte, 'polnische' Notation ist ein anderes FORTH Hindernis. Wer bereits mit HP-Taschenrechnern gearbeitet hat, wird wissen was ich meine. Für alle, die UPN nicht kennen ein kleines Beispiel, natürlich in Gestalt eines Programms.

In BASIC:	In FORTH:
For I=0 to 2000	:Bench
F=10/3	0.
Next I	2000 0 Do
Print F	Drop Drop 10. 3. F/
	Loop
	F.;

Alles klar? Wahrscheinlich nicht, aber das macht nichts. Es ist tatsächlich enorm schwer, sein erstes lauffähiges FORTH Programm zu schreiben. Das liegt mit Sicherheit auch an der neuen Eingabetechnik UPN. Also zweifeln Sie nicht an Ihrem Verstand. Wer am Ball bleibt wird sein hart erkämpftes FORTH später nicht wieder hergeben.

Eigenarten

Das uns vorliegende TEX FORTH unterstützt alle TI-Funktionen. Angefangen bei der Erstellung eigener Grafiken bis zum Ansprechen der Schnittstellen, weist das TEX FORTH alle komfortträchtigen Eigenschaften unserer TI Konsole auf und darüberhinaus noch erheblich mehr. Alle einmal definierten Befehlswörter werden zum festen Bestandteil des Betriebssystems, bis der Rechner mit dem Befehl FORGET dazu gezwungen wird, bestimmte Befehle wieder zu vergessen. Wer aber genügend Disketten sein Eigen nennt, wird dies wohl kaum machen. Mit TEX FORTH läßt sich direkt der sagenhafte Bit Map Mode ansprechen. Der dann folgende rasante Bildaufbau spricht für sich selbst. Selbstverständlich können Sie sofort wieder in den Textmode umschalten, TEX FORTH macht es möglich. Und wer Lust hat, der kann sich seinen eigenen bildschirmorientierten Editor erstellen. Die Möglichkeiten von TEX FORTH sind nahezu unerschöpflich. Nur müssen Sie sich in diese Sprache intensiv hineinarbeiten. Sonst erleiden Sie sofort Schiffbruch. Und das ist keinem User zu wünschen.

TEX FORTH erfordert eine Gerätekonfiguration, die leider nicht jeder User besitzt. Sie benötigen ein Diskettenlaufwerk, das Editor Assembler Modul und die Memory Expansion. Dann kann es losgehen. Die FORTH Diskette wird von einem umfangreichen Manual begleitet, das unbedingt zu lesen ist. Leider ist dieses Handbuch zur Zeit nur in englischer Sprache zu haben. Laut Computer Hüsli soll in Kürze eine deutsche Übersetzung folgen. Der Preis für diese Übersetzung soll bei etwa 100 Mark liegen. Ein stolzer Preis. Hoffen wir, daß die Qualität der Übersetzung hält, was der Preis verspricht. Und dann sollten Sie sich das Handbuch des Editor Assemblers zur Brust nehmen. Wissen ist schließlich Macht. Wem das Handbuch mit seinen 400 Seiten zu viel Stoff ist, den möchten wir an dieser Stelle auf unsere Peek und Poke Serie für den TI hinweisen. Das dort vermittelte Wissen über Systemadressen unseres TIs läßt sich direkt beim Erlernen von FORTH anwenden. Darüber hinaus ist es empfehlenswert, sich etwas Literatur über FORTH zuzulegen, zumal der Einstieg in diese Sprache nicht vom Manual unterstützt wird.
Hier eine Titelauswahl von Büchern, die uns besonders empfehlenswert erscheinen:

Starting FORTH
von Leo Brodie

Using FORTH
von FORTH Inc.

Invitation to FORTH
von Katzan

Wer noch mehr wissen will, kann sich auch direkt an die FORTH Interest Group wenden.

Adresse:
FORTH Interest Group
P.O. Box 1105
San Carlos, CA 94070
USA

Fazit

Forth, besonders das uns vorliegende TEX FORTH, stellt eine enorme Erweiterung für den TI dar. Die extreme Vielseitigkeit und Ausbaufähigkeit von Forth verleiht dem TI Möglichkeiten, die manch größeres System nicht eingebaut hat. Wir können TEX FORTH wärmstens empfehlen. Anfänger sollten allerdings die Finger von FORTH lassen, denn die Anforderungen an den Programmierer sind erheblich. Wer aber über Mut und über entsprechende Computerkenntnisse verfügt, der sollte FORTH probieren. Der absolute Höhenflug ist ihm sicher.

Klaus Weidemann

Bücher für Computerfans

Maschinensprache für Spectrum

Früher oder später steht jeder engagierte User vor der Frage: Wie komme ich an die Maschinenprogrammierung heran? Wege gibt es viele. Eine knapp gefaßte, preisgünstige und trotzdem recht informative Möglichkeit ist das 'Maschinencode-Handbuch für den ZX-Spectrum' von Profisoft.

Der Z80 wird, auch in Details wie den einzelnen Registern, der Kontrolleinheit oder dem Programmzähler, kurz beschrieben. Die Z80-Maschinencodebefehle sind aufgeführt und kommentiert. Einzelne Vorgänge, wie das Laden eines Registers in ein anderes oder Unterprogrammaufrufe werden an Beispielen demonstriert.

Im Anschluß finden sich ROM-Adressen und kurze Anwendungen in Maschinencode wie das Drucken von Zeichenketten, PLOT, BEEP. Teil 2 bringt kurze Informationen über BASIC, zu wenig, um sinnvoll zu sein. Im Anhang finden wir unter anderem die BASIC-Befehle mit ihren Einsprungadressen und Befehlsparametern, die Speicherorganisation und weitere Z80-Befehle sowie Hinweise auf ROM-Fehler.

Merz
Maschinencode Handbuch für den ZX Spectrum
Verlag profisoft
77 Seiten
DM 25,–
ISBN 3-9239 85-02-9

Gemischtes Doppel

Wieso jemand darauf kommt, aus allen verfügbaren Home-Computern ausgerechnet den TI 99/4A und den VC 20 zu einem Team zusammenzukoppeln und ein gemeinsames Textverarbeitungsprogramm zu entwickeln, erscheint uns rätselhaft. Doch für diese beiden Rechner ist tatsächlich eine recht interessante Textverarbeitung entstanden, die sich zudem noch mit einfacher Peripherie begnügt. Die Speicherung erfolgt auf Cassette.

Der Leser, der unvorbelastet durch BASIC-Kenntnisse das Buch aufschlägt, findet in Vorwort und Vorbemerkung die wesentlichen Befehle, speziell zu den String-Operationen. Auch die Drucker-Steuerung wird erläutert.

Im dritten Kapitel geht's in die Vollen: generelle Überlegungen zur Programmierung eines kompletten Textsystems. Stück für Stück folgen Steuerungsmodul und Unterprogramme, die alle sehr gut lesbar sind. Doch Vorsicht — auf jeden Fall den begleitenden Text aufmerksam lesen, dann erkennt man einige der Fehler in den Listings.

Das Programm liefert Speicher- und Ladefunktionen, läßt Editieren und Formatieren und steuert natürlich das Drucken.

Wir halten das Buch und das Programm für recht gelungen, da auch dem Fortgeschrittenen etwas geboten wird.

Tölke
Textverarbeitung mit Programm für TI 99/4 A und VC 20
Verlag Vieweg
138 Seiten
DM 36,–
ISBN 3-528-04276-1

Die schnelle Mark per Rechner

Was war ich dumm! Glaubte, mit meinem kleinen Monatsgehalt auskommen zu müssen! Habe Lotto gespielt im Glauben, damit mal das große Glück zu machen. Bin förmlich über das Geld, das nur so auf der Straße lag und mich anlachte, hinweggestolpert.

Doch nun weiß ich, ein Computer ist eine Geldmaschine! Nicht, daß ich mir jetzt mit dem Matrixdrucker die Riesen ausdrucke oder aus dem Schlitz der Floppy die Heiermänner rauspurzeln. Aber man kann ja soo viel Geld mit dem Computer verdienen — man glaubt es kaum. Im Buch "Die Geldmaschine" von Joe Weisbecker steht es schließlich schwarz auf weiß. Jede einzelne der 250 Ideen für lukrative Nebenverdienste mit dem Mikrocomputer werde ich ausführen: schon morgen eine Erfindung machen (die Adresse des Deutschen Patentamtes steht auch drin), die von einer Firma für gutes Geld ganz bestimmt gekauft wird.

Übermorgen ein Buch schreiben und erfolgreich vermarkten (falls sich kein Verlag findet: den gründet der Erfolgreiche schnell selbst). Für die restlichen Tage der Woche bleibt noch genug zu tun. Programme schreiben (aber nur solche, die leicht zu verkaufen sind), einen Versandhandel oder Geräteverleih aufziehen (die paar Tausender Startkapital sind ja wohl ein Klacks). Eines muß man eben ständig haben: Ideen!

Mein Freund, ein ganz pessimistischer und phantasieloser Bursche sagt, in dem Buch seien zwar unheimlich viele Ideen beschrieben, aber nur so oberflächlich, daß man die eigentlichen Probleme gar nicht auf Anhieb erkennt. Daß Fragen der Finanzierung, der Abwicklung (z.B. beim Versandgeschäft) oder urheberrechtlicher Art einfach vernachlässigt werden. Der sagt sogar, das ganze Buch sei mit seinem Preis von knapp 50 Mark reine Geldschneiderei.

Herr Weisbecker meint zu diesem Problem:

"Übrigens besteht das Geheimnis, reich zu werden, nur darin, daß Sie 'Das Geheimnis, reich zu werden' für 100 DM an andere Leute weiterverkaufen."

Weisbecker
Die Geldmaschine
Verlag Norman Rentrop
225 Seiten
DM 48,–
ISBN 3-8125-0036-1

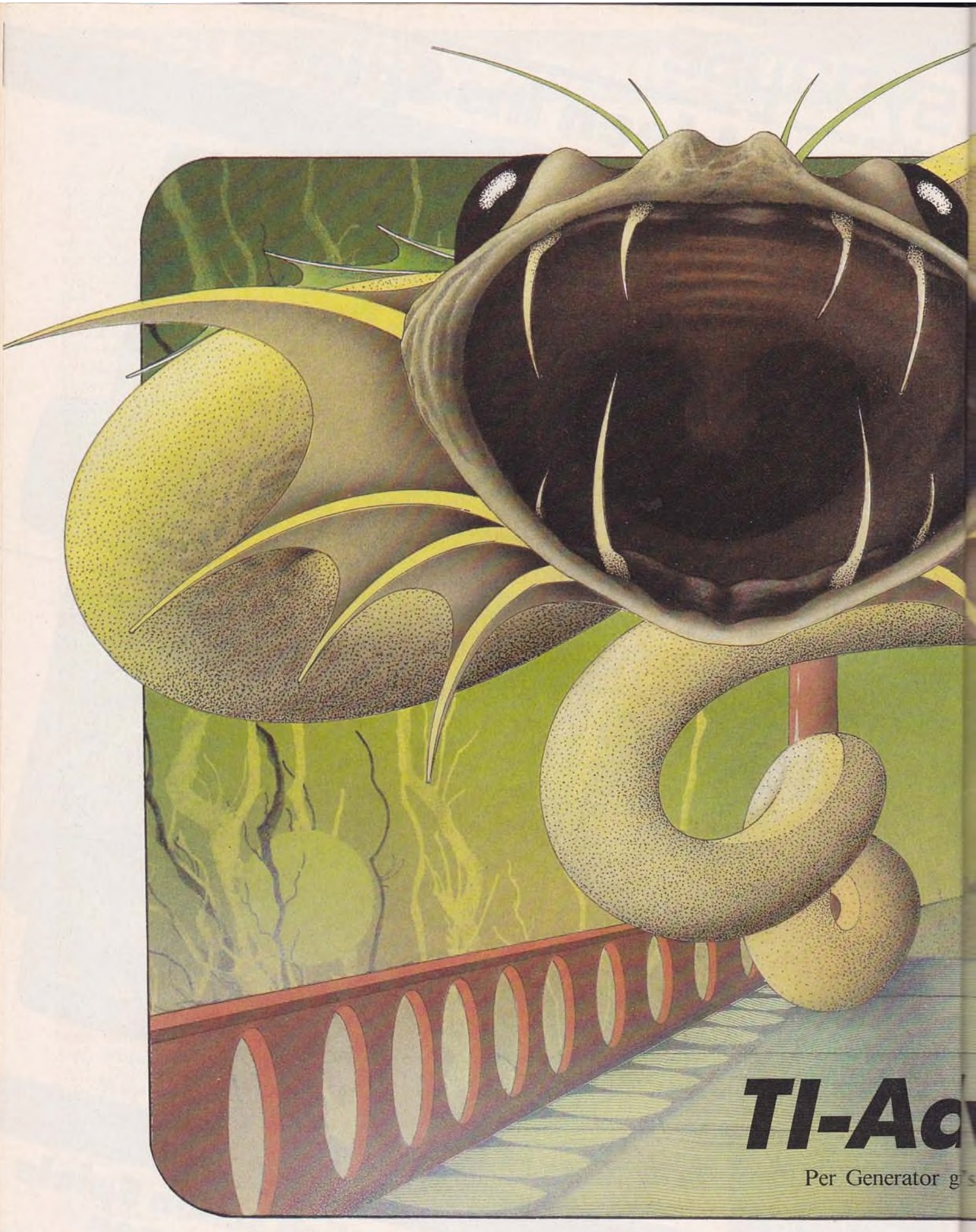
Das 600XL/800XL Handbuch

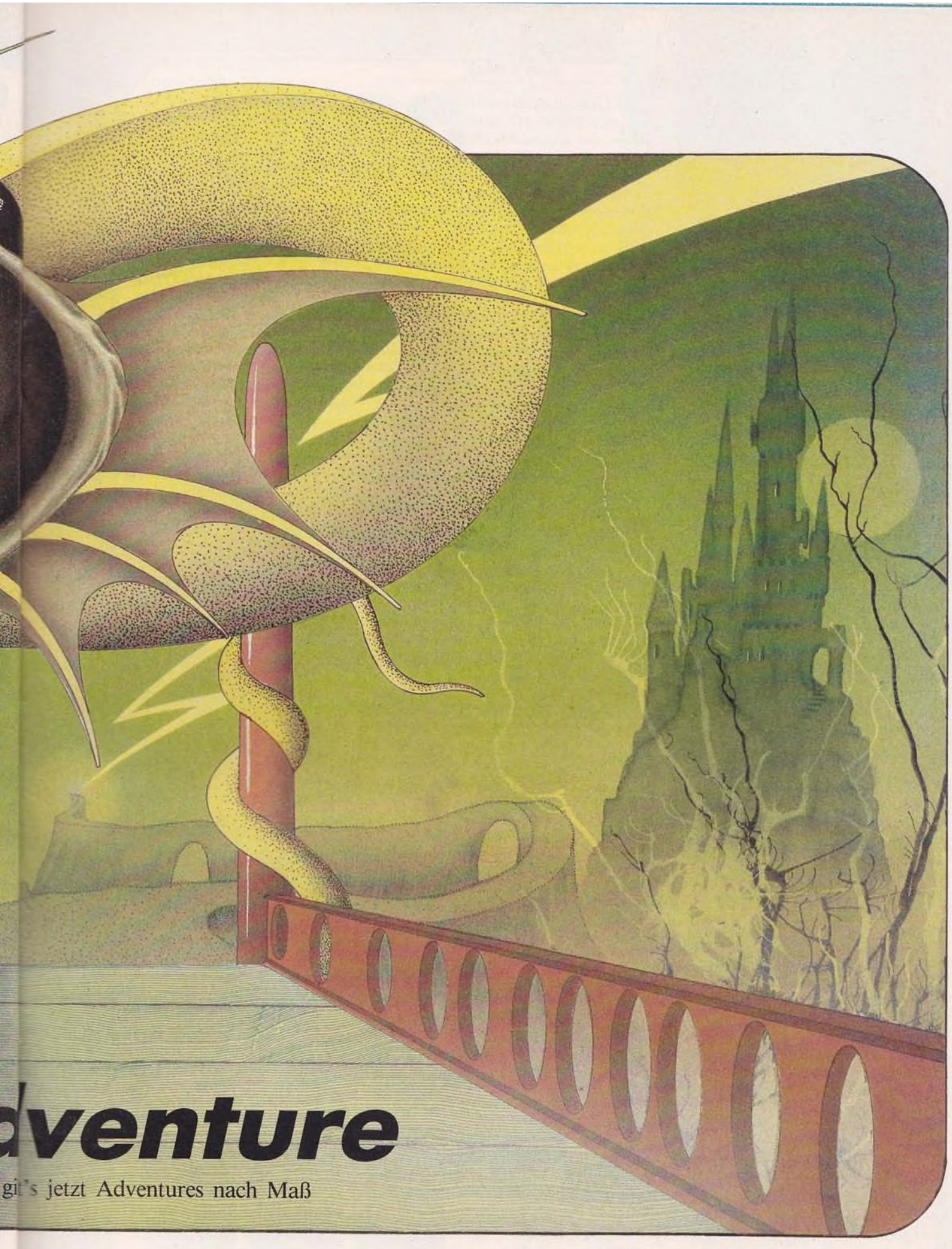
Das Atari BASIC-Reverence-Manual ist seit einiger Zeit nicht mehr erhältlich, und so gab es außer den BASIC-Karten kaum gedruckte Unterstützung für Einsteiger.

In dem neuen Handbuch werden viele Texte des alten Manuals verwendet und auf die Geräte der XL Serie abgestimmt. Von berufener Seite wird der Anfänger mit den Besonderheiten des Atari-BASIC vertraut gemacht. Diese Einführung in BASIC ist so ausführlich, daß man fast von einem BASIC-Kurs sprechen kann.

Zu jedem Befehl werden Beispiele gegeben, die auch von einem Newcomer verstanden werden. 15 Übungs-Beispiele vertiefen die erworbenen Kenntnisse und geben dazu einige Programmiertricks. Die gut erklärten Fehlermeldungen und eine Liste der wichtigsten Systemadressen helfen bei den ersten Programmier-Versuchen. Ein Literatur-Verzeichnis über den Atari und die Pin-Belegung der Peripherie-Anschlüsse zeigen uns, daß Atari mehr für den User tun möchte. Wir meinen: weiter so!

Das 600XL/800XL Handbuch,
96 Seiten DIN A 4,
ATARI Katalog-Nr. 91110. Schutzgebühr 4.50 Mark


TI-Ac
Per Generator g



dventure

gi's jetzt Adventures nach Maß

TI-Adventure

Die Spannung eines Adventure-Games hängt immer vom Witz des Spielverlaufs ab. Viele der im Handel erhältlichen Abenteuerspiele sind teuer und langweilig, im Gegensatz zu unseren in Heft 12/84 veröffentlichten TI-Adventure. Wesentlicher Unterschied zu anderen Programmen: Es handelt sich dabei um ein Rahmenprogramm, das verschiedenste Spielabläufe gestattet.

Zu diesem Rahmenprogramm brachten wir ein weiteres, das Ihnen auf Diskette oder Cassette, je nach Geschmack, eine Datei erstellte, die dem Spiel erst die Würze gab.

Hier haben wir nun den Editor zum Abenteuer. Das Rahmenprogramm wird weiterverwendet, aber mit dieser Ergänzung können Sie nun nach Herzenslust Ihre eigenen Adventures editieren.

Daß das nicht ganz so einfach ist, liegt auf der Hand. Doch mit dem Editor erspart sich der Spielbegeisterte viel mühsames Zuordnen von Daten in bestimmte Daten-Sätze.

Wie erstellt man nun sein Adventure? Alles fängt damit an, daß man sich eine Handlung einfallen läßt. Möglichst logisch sollte sie sein und in mehrere Situationen unterteilt werden. Jeweils acht Situationen werden dann zu einem Block mit einem Eingang in die Situation '0' und einem Ausgang aus Situation '7' zusammengefaßt. Situation '7' kann aber auch das Ziel der Reise sein.

Wichtig ist dabei, daß die Situationen jeweils vier Wege zu vier anderen haben können, aber nicht müssen. Zum Beispiel kann man jemanden in die Irre führen, indem man einen Weg in eine Situation hinein — aber keinen herausführen läßt.

Unsere Grafik zeigt Ihnen mit Hilfe der Verbindungslinien, auf welche Weise die einzelnen Situationen, hier die Ecken des Würfels, miteinander verbunden werden können. Manche der Wege sind Einbahnstraßen, auf anderen gibt es Probleme.

Am besten macht man sich von jedem Block eine Zeichnung, um sich merken zu können, welche Wege gesetzt werden müssen. Ihrer Fantasie können Sie so völlig freien Lauf lassen.

Zum Listing

Die Zeilen 10 bis 70 gehören nicht zum eigentlichen Editor. Mit dieser Kurz-Routine werden die numerischen Datas jeweils zu acht addiert, und die Zahl auf den Bildschirm gebracht. Die korrekten Werte finden Sie in der REM-Zeile 60. Wenn die Zahlen übereinstimmen, löschen Sie die Zeilen 10 bis 70 einfach wieder.

Die REM-Zeilen können im ganzen Programm fortgelassen werden, sind aber eine wichtige Hilfe bei der Suche nach Tippfehlern.

140—160 Daten. Sie werden in die Arrays RI$, SO und SO$ eingelesen.

220/230 Hier suchen Sie sich wieder die Zeile aus, die Sie brauchen. Zeile 220 nur für Cassetten-System, Zeile 230 nur für Disketten-System. Das REM-Statement am Anfang der Zeile muß denn entsprechend herausgenommen werden. Beide Zeilen eröffnen die Adventure-Datei, für Diskette getrennt in HIGH (für den Highscore) und ADDAT (für die Spiel-Daten).

270—310 Dimensionierung der Arrays, laden der Datas.

350—450 Der Editor fragt für jede Situation die Wege ab und erkundigt sich, welche Probleme gesetzt werden sollen.

490—630 Der nächste Schritt ist die Beschreibung der Situationen. Keine sollte mehr als vier Bildschirmzeilen lang sein, da sie sonst nicht mehr in die Spielmaske paßt. Gegenstände, die man mitnehmen kann, und die Namen der Wege sollten nur mit Großbuchstaben eingegeben werden, da es sonst zu unerwünschten Fehlfunktionen im Spiel kommen kann.

660—770 Alle eingegebenen Daten werden noch einmal Situation für Situation auf den Bildschirm gebracht. Jeder Gruppe steht eine Ziffer voran. Drückt man diese Ziffer, so können die dazugehörigen Daten korrigiert werden. Die '0' schließt den Block.

810—860 In diesen Zeilen darf man sich auf gar keinem Fall einen Tippfehler erlauben; die Daten werden auf Band bzw. Diskette geschrieben.

900—910 Wurde für Situation '7' bei der Frage, ob das Ziel erreicht sei, eine '1' eingegeben, so wird die Datei geschlossen. Das neue Adventure kann dann gespielt werden. Sonst springt der Rechner zurück in die Zeile 350, und der nächste Block kann eingegeben werden.

950—1090 Diese Routine erlaubt die Korrektur der eingegebenen Daten. Bis auf den Situationstext werden alle Daten direkt in der Maske korrigiert. Anschließend werden die Daten erneut auf den Bildschirm gebracht.

Bei der Eingabe bekommt ein 'Typ' Problem einen Namen. Hierbei handelt es sich um einen Gegenstand, den man ablegen muß. Folglich muß er auch irgendwo zum Mitnehmen bereitliegen.

Wir sind sicher, daß Sie und Ihre Freunde mit dem Adventure-Programm und diesem Editor viel Freude haben werden. Besonders gute Spiele wollen wir dann in den nächsten Heften in der Form eines Daten-Generators wie in Heft 12/84 veröffentlichen.

Und sollte jemand einen besseren Editor entwickelt oder gar das Adventure-Programm verbessert haben, so würden wir uns auch darüber freuen, diese Entwicklungen an den Rest der Leserschaft weitergeben zu können.

Anwenderfreundlich wartet der Editor auf die wildesten Fantasien.

Die Variablen:

Arrays:	Der Editor arbeitet mit der Basis 1 für Feld-Variable.
TEXT$(8)	Situations-Beschreibungen
SO$(5)	Hilfsarray für die Probleme
L$(8,4)	Gegenstände zum Mitnehmen und das abzulegende Problem
RI$(8,4,3)	Die möglichen Richtungen, ob gesetzt und deren Bezeichnung
COL(8)	Bildschirmfarbe der Situation
SO(8,5,2)	Die fünf möglichen Probleme, ob und auf welchem Weg sie gesetzt wurden

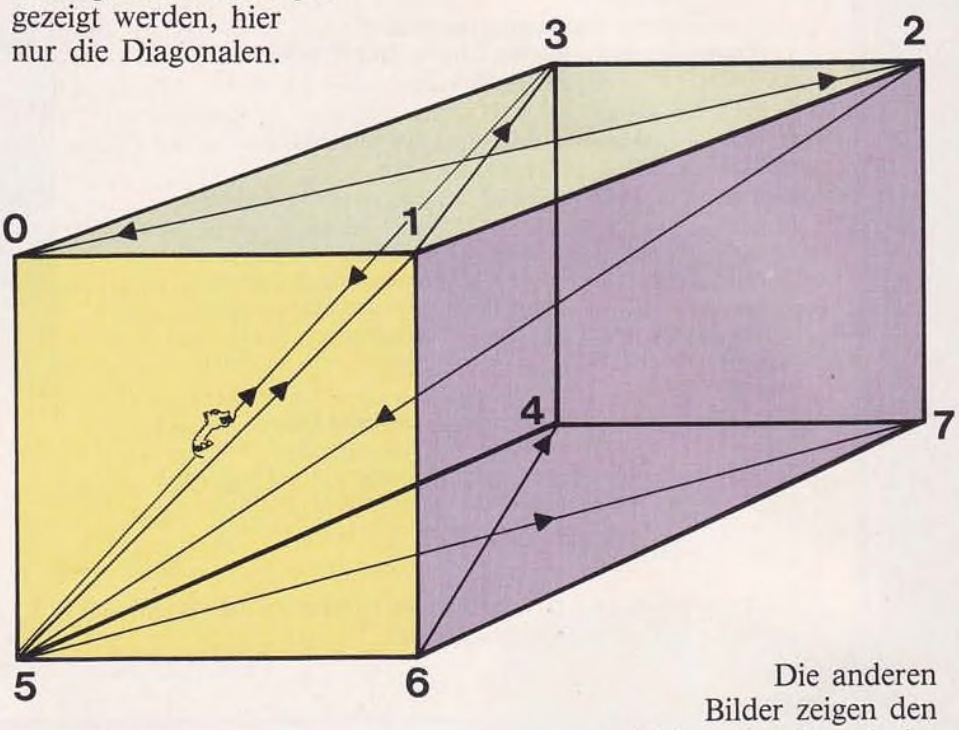
Arbeits-Variable:

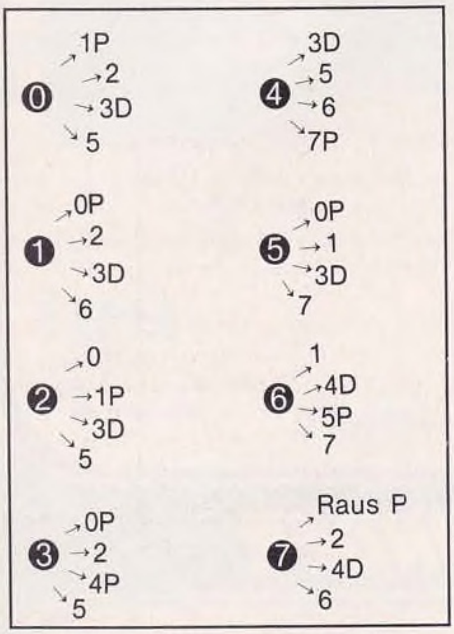
X$	Abfrage
K,S	Tastaturabfrage
ZAE	Block-Zähler
ZIEL	Definiert Situation '7' als Ziel
I,J	FOR/NEXT-Schleifen

Klaus Weidemann

Es ist natürlich etwas schwierig, die Zusammenhänge der verschiedenen Situationen im Wort zu erklären. Zu diesem Zweck stehen diese Abbildungen auf den Seiten.
Das große räumliche Bild zeigt Ihnen, wie die einzelnen Situationen miteinander zusammenhängen, wobei nur die wichtigsten Beziehungen gezeigt werden, hier nur die Diagonalen.

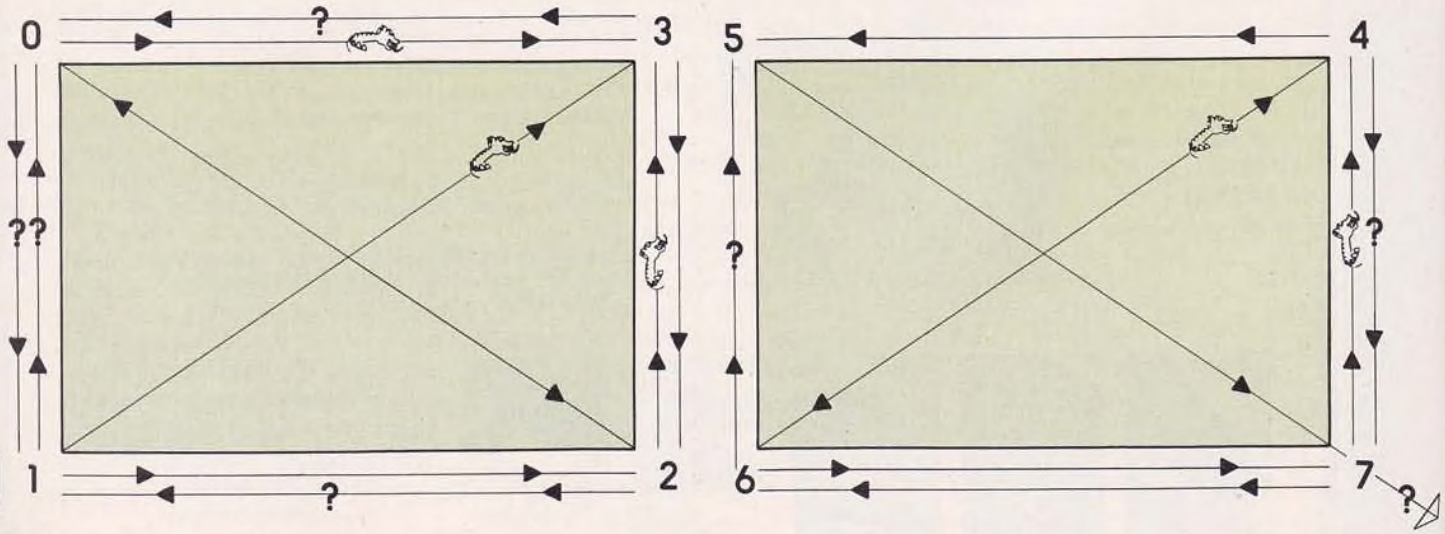
TI-Adventure

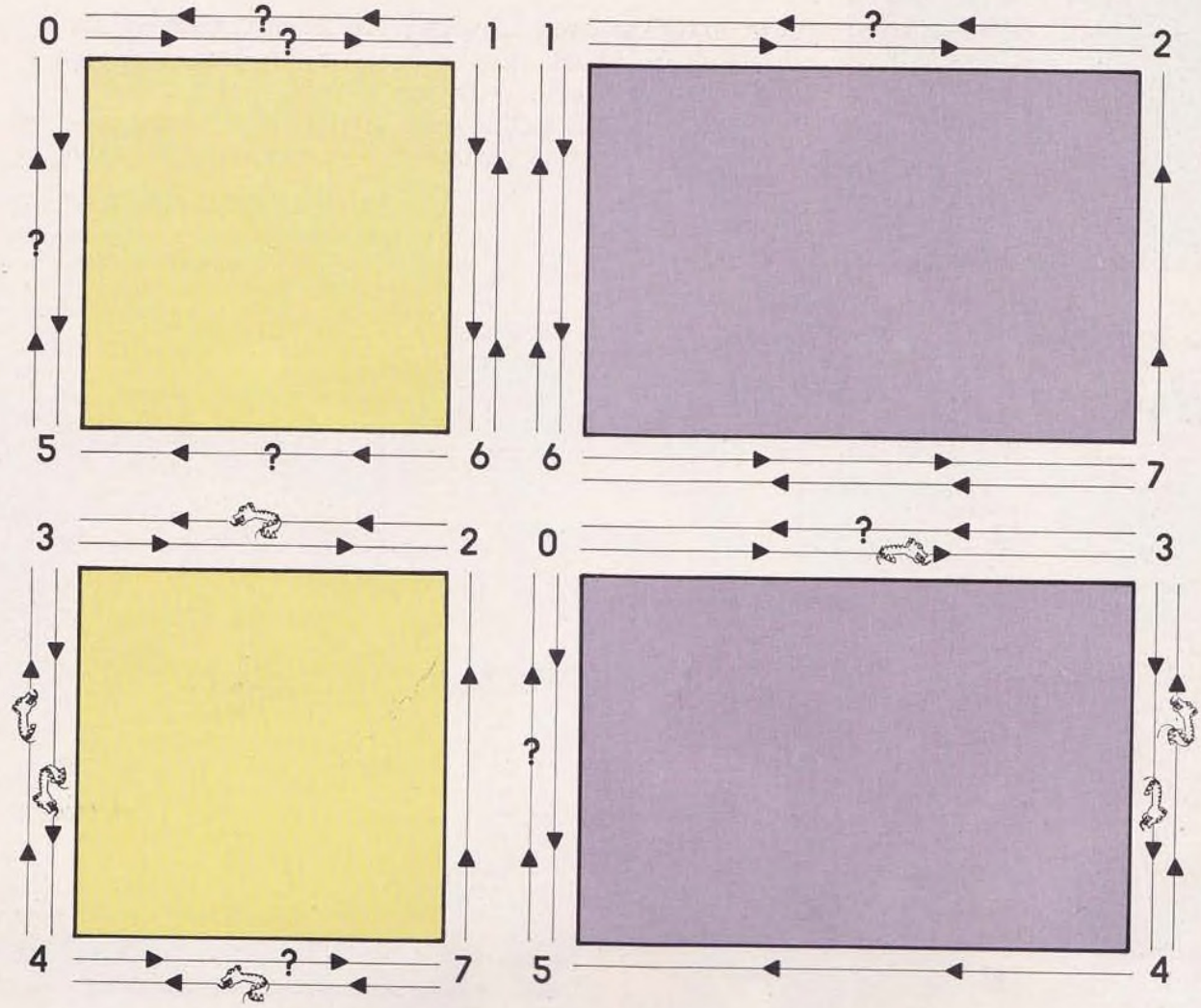




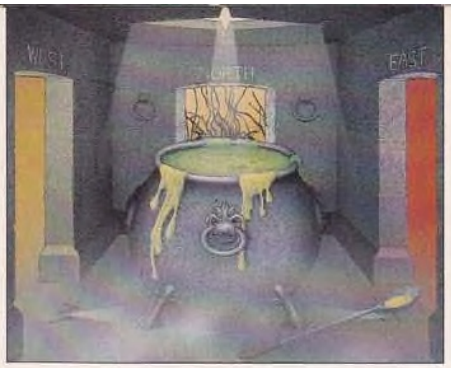
Von Situation ❶ kommt man → mit Problem (P) nach 0, über Drachen (D) nach 3

Die anderen Bilder zeigen den gleichen Quader wie im ersten Bild, nur haben wir ihn der Deutlichkeit halber total zerlegt. Jede Deckfläche wird einzeln mit den entsprechenden Verbindungen zu anderen Situationen aufgezeigt. Die komischen Zahlen an den Ecken sind die jeweiligen Situationen oder auch Bilder im Adventure. Natürlich kann es beim Sprung in eine andere Situation zu bösen Zwischenfällen kommen. Dafür bürgt der kleine Drache, der sich Ihnen manchmal in den Weg stellt oder das Fragezeichen, das Ihnen einige Probleme bereiten soll. Die einzelnen Deckflächen zeigen Ihnen alle gangbaren Verbindungen mit anderen Situationen auf. Wir hoffen, Ihr Gesamtverständnis für das Adventure gefördert zu haben.






```
 10 ! CHECKSUM
 20 !
 30 FOR I=1 TO 6 :: B=0 :: FOR J=1 TO 8 :: READ A ::
     B=A+B :: NEXT J :: PRINT B;:: NEXT I :: PRINT
 40 STOP
 50 !
 60 ! Korrekte Werte: 22, 20, 32, 37, 15, 34
 70 !
100 ! Alle REM's koennen raus
110 !
120 ! Daten-Saetze fuer Richtung, Feinde und
     Besonderheiten
130 !
140 DATA 1,2,3,5,0,2,3,6,0,1,3,5,0,2,4,5,3,5,6,7,0,
    1,3,7,1,4,5,7,8,2,4,6
150 DATA 1,3,0,3,1,3,0,4,3,7,0,3,5,4,8,4
160 DATA Problem,Feind,Drache,Vogel,Dunkel
170 !
180 ! Zeile 220 nur fuer Cassette Zeile 230 nur fuer
     Diskette
190 !
200 ! Datei-Eroeffnung
210 !
220 REM OPEN #1:"CS1",OUTPUT,INTERNAL,FIXED 192 ::
     PRINT #1:0
230 REM OPEN #1:"DSK1.HIGH",OUTPUT,INTERNAL ::
     PRINT #1:0 :: CLOSE #1 :: OPEN #1:
    "DSK1.ADDAT",INTERNAL,OUTPUT,FIXED 192
240 !
250 ! Initialisierung, Dimensionierung und einlesen
     der Datas
260 !
270 ON WARNING NEXT :: ZAE=1 :: OPTION BASE 1 ::
     DIM TEXT$(8),COL(8),L$(8,4),SO$(5),SO(8,5,2),
    RI$(8,4,3)
280 CALL CLEAR :: DISPLAY AT(1,3):
    "ADVENTURE-DATENGENERATOR"
290 FOR I=1 TO 8 :: FOR J=1 TO 4 ::
     READ RI$(I,J,1):: NEXT J :: NEXT I
300 FOR I=1 TO 8 :: READ SO(I,1,1),SO(I,3,1)::
     NEXT I
310 FOR I=1 TO 5 :: READ SO$(I):: NEXT I
320 !
330 ! Eingabe der Wege, Besonderheiten und ob das
     Ziel erreicht ist
340 !
350 FOR I=1 TO 8 :: DISPLAY AT(1,3):"Block";ZAE
360 DISPLAY AT(3,3):"Situation";I-1;"kann nach:"
370 FOR J=1 TO 4 :: DISPLAY AT(5+2*J,2):
    "Sit ";RI$(I,J,1);TAB(11);
    "J" :: NEXT J :: DISPLAY AT(15,2):"fuehren,": :
    "<J> fuer Weg setzen": :
    "<N> fuer Weg nicht setzen"
380 FOR J=1 TO 4 :: ACCEPT AT(5+2*J,11)BEEP SIZE(-1)
```



TI-Adventure

```
    VALIDATE("JN"):RI$(I,J,2):: NEXT J
390 DISPLAY AT(21,1):"Weg";SO(I,1,1): :
    "KANN ein Problem haben <0/1>":
    TAB(25);"0" :: ACCEPT AT(24,25)SIZE(-1)
    VALIDATE("01"):SO(I,1,2)
400 DISPLAY AT(21,4):SO(I,3,1): :
    "KANN gesperrt sein <0/1>":
    TAB(25);"0" :: ACCEPT AT(24,25)SIZE(-1)
    VALIDATE("01"):SO(I,3,2)
410 DISPLAY AT(21,1):"Gibt es Feinde? <0/1>";0: : :
     : :: ACCEPT AT(21,23)SIZE(-1)VALIDATE("01"):
    SO(I,2,2)
420 DISPLAY AT(21,1):"Singt ein Vogel? <0/1>";
    0 :: ACCEPT AT(21,24)SIZE(-1)VALIDATE("01"):
    SO(I,4,2)
430 DISPLAY AT(21,1):"Ist es Dunkel? <0/1>";
    0 :: ACCEPT AT(21,22)SIZE(-1)VALIDATE("01"):
    SO(I,5,2)
440 NEXT I
450 DISPLAY AT(21,1):"Ist das Ziel erreicht? <0/1>"
    TAB(25);"0" :: ACCEPT AT(22,25)SIZE(-1)
    VALIDATE("01"):ZIEL
460 !
470 ! Eingabe der Texte fuer Situation,
     Gegenstaende, Wege und Probleme, sowie
     Bildschirmfarbe
480 !
490 FOR I=1 TO 8
500 DISPLAY AT(10,3)ERASE ALL:"TEXT-EINGABEN"
510 DISPLAY AT(12,3):"Situations-Beschreibung:": :
    I-1
520 LINPUT TEXT$(I)
530 PRINT : : :"Bildschirm-Farbe" ::
     INPUT "nach Tabelle: ":
    COL(I):: IF COL(I)<3 OR COL(I)>16 THEN PRINT :
    "falsche Eingabe": : :: GOTO 530
540 PRINT : :"Liegt was rum? <J/N> J";
    :: ACCEPT AT(24,22)SIZE(1)VALIDATE("JN"):X$
550 IF X$="N" THEN 580
560 FOR J=1 TO 3 :: PRINT : :"Gegenstand";J:
    "Leerstring fuer nichts" :: INPUT "":L$(I,J)
570 NEXT J
580 IF SO(I,1,2)=1 THEN PRINT : :
    "Diese Situation hat ein":
    "Problem." :: INPUT "Wie heisst es? ":L$(I,4)
590 FOR J=1 TO 4
600 IF RI$(I,J,2)="N" THEN 620
610 PRINT : :"Ein Weg fuehrt nach ";RI$(I,J,1):
     :: INPUT "Wie heisst er? ":RI$(I,J,3)
620 NEXT J
630 NEXT I
640 !
650 ! Ausgabe zur Ueberpruefung der eingegebenen
     Daten
660 !
670 FOR I=1 TO 8
680 CALL CLEAR
690 DISPLAY AT(1,1):"1. Text";I-1;TAB(12);
    "2. Farbe =";COL(I): :
    TEXT$(I):: CALL SCREEN(COL(I))
700 DISPLAY AT(9,1):"3.":"zum Mitnehmen" ::
     FOR J=1 TO 3 :: DISPLAY AT(10+J,1):
    L$(I,J):: NEXT J
710 DISPLAY AT(15,1):"4. Problem";SO(I,1,2):L$(I,4):
     :"5. Richtungen"
720 FOR J=1 TO 4 :: DISPLAY AT(18+J,2):
    RI$(I,J,1);" ";RI$(I,J,2);" ";
    RI$(I,J,3):: NEXT J
730 DISPLAY AT(9,15):"6. Besonderes" ::
     FOR J=2 TO 5 :: DISPLAY AT(9+J,15):
    SO$(J):: DISPLAY AT(9+J,23):SO(I,J,2):: NEXT J
740 IF I=8 THEN DISPLAY AT(18,15):"7. ZIEL =";ZIEL
750 DISPLAY AT(24,1):"0=Okay,   Ziffer=Aendern";
760 CALL KEY(0,K,S):: IF S=0 OR K<48 OR K>55
    THEN 760 ELSE IF K>48 THEN 950
770 NEXT I
780 !
790 ! Abspeichern des Daten-Blocks auf Cassette oder
     Diskette
800 !
810 CALL CLEAR :: PRINT "Block";ZAE;
    "wird geschrieben": :
820 FOR I=1 TO 8 :: PRINT #1:
    TEXT$(I),COL(I):: NEXT I
830 FOR I=1 TO 8 :: PRINT #1:
    L$(I,1),L$(I,2),L$(I,3),L$(I,4),SO(I,1,2),
    SO(I,2,2),SO(I,3,2),SO(I,4,2),SO(I,5,2):: NEXT I
840 FOR I=1 TO 5 STEP 2
850 PRINT #1:RI$(I,1,3),RI$(I,2,3),RI$(I,3,3),
    RI$(I,4,3),RI$(I+1,1,3),RI$(I+1,2,3),
    RI$(I+1,3,3),RI$(I+1,4,3):: NEXT I
860 PRINT #1:RI$(7,1,3),RI$(7,2,3),RI$(7,3,3),
    RI$(7,4,3),RI$(8,1,3),RI$(8,2,3),RI$(8,3,3),
    RI$(8,4,3),ZIEL
870 !
880 ! Schliessen der Datei bei ZIEL=1
890 !
900 IF ZIEL=1 THEN CLOSE #1 ::
     PRINT "Das neue Adventure ist": :
    ZAE;"Bloecke mit je": :
    "8 Situationen lang." :: STOP
910 ZAE=ZAE+1 :: CALL CLEAR :: GOTO 350
920 !
930 ! Korrektur-Routine
940 !
950 ON K-48 GOTO 960,980,1000,1010,1040,1080,1090
960 CALL CLEAR :: PRINT "Situations-Text";I-1: :
    TEXT$(I): :"Bitte den Neuen Text eingeben:": :
970 LINPUT TEXT$(I):: GOTO 680
980 ACCEPT AT(1,22)BEEP VALIDATE(DIGIT)SIZE(-2):
    COL(I)
990 IF COL(I)<3 OR COL(I)>16 THEN 980 ELSE 680
1000 FOR J=1 TO 3 :: ACCEPT AT(10+J,1)BEEP SIZE(-13):
    L$(I,J):: NEXT J :: GOTO 680
1010 ACCEPT AT(15,12)BEEP SIZE(-1)VALIDATE("01"):
    SO(I,J,2)
1020 IF SO(I,J,2)=1 THEN ACCEPT AT(16,1)
    BEEP SIZE(-13):L$(I,4)
1030 GOTO 680
1040 FOR J=1 TO 4 :: ACCEPT AT(18+J,5)BEEP SIZE(-1)
    VALIDATE("JN"):RI$(I,J,2)
1050 IF RI$(I,J,2)="N" THEN RI$(I,J,3)="" ::
     GOTO 1070
1060 ACCEPT AT(18+J,7)SIZE(-21)BEEP:RI$(I,J,3)
1070 NEXT J :: GOTO 680
1080 FOR J=2 TO 5 :: ACCEPT AT(9+J,23)SIZE(-1)
    VALIDATE("01")BEEP:SO(I,J,2):: NEXT J
1090 IF I<8 THEN 670 ELSE ACCEPT AT(18,22)SIZE(-1)
    VALIDATE("01"):ZIEL :: GOTO 680
```

Ein Stadtteil greift zum

Computer

Böse Zungen behaupten, Bremen sei ein Vorort von Hamburg. Computermäßig, das ist sicher, dürfte der kleinste Bundesstaat Deutschlands zumindest für alle die, die nicht in unmittelbarer Nähe wohnen, einen weißen Fleck in ihrer imaginären Landkarte darstellen. Für uns war das nicht anders, zugegeben. Doch mußten wir einige angenehme Überraschungen einstecken, als wir der Einladung der 1. Bremer User Group nach Obervieland Folge leisteten.

Mit dem Namen 'Group' fangen die Besonderheiten schon an, die in Bremen auf Computerfans warten. Man trifft sich nämlich in lockerer Form. Wer gerade Lust und Zeit hat, der kommt am Donnerstagabend in den Medienraum des Zentrums, eines großzügig angelegten Gebäudes, das allen Bewohnern des Stadtteils offen steht. Einen Beitrag bezahlen muß er nicht, auch keine Vereinsmeierei betreiben. Spaß am Computern und am Umgang mit Leuten, das sind die einzigen Voraussetzungen, derer es bedarf, um hier klarzukommen.

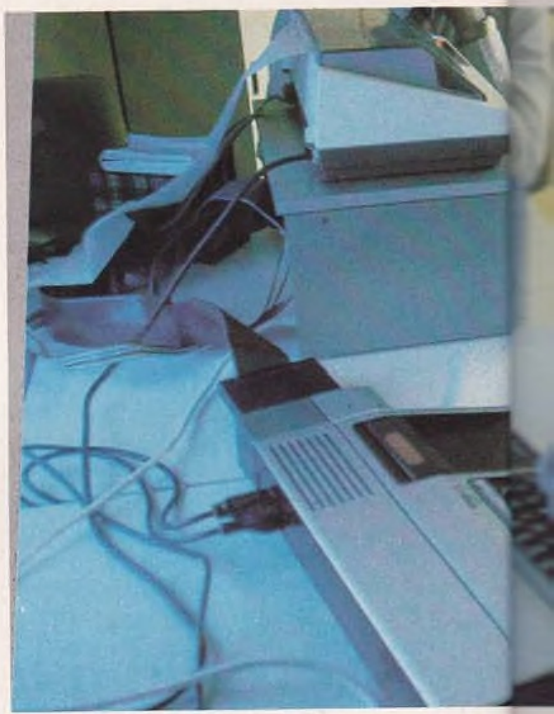
Die Idee der Stadtteilzentren, in Bremen 'Bürgerhäuser' genannt, baut darauf, daß all denen ein Treffpunkt geboten wird, die in ihrer Freizeit einem Hobby nachgehen, das gemeinsam mehr Spaß bringt. Seit das Bürgerhaus in Obervieland im Mai 1977 seine Pforten öffnete, haben viele Bremer die Vorteile einer solchen Einrichtung schätzen und nutzen gelernt. Gleich, ob man die Disco-Gruppe nimmt, die, ohne jemanden zu stören, die Wände wackeln läßt und den Jugendlichen des Stadtteils hilft, mit ihrem Taschengeld besser über die Runden zu kommen, ob man sich die Foto-Freaks als Beispiel wählt, die eine gut ausgerüstete Dunkelkammer benutzen können, oder die Holzwerker — alle finden hier nebeneinander Platz, um ihren Interessen ungestört nachgehen zu können. Und nicht nur nebeneinander: Hier kommt es nicht zu Clubs und Clübchen, die schier gar nichts voneinander wissen. Denn die Aufgabe der Gestaltung des Hauses selbst ist in die Hände der Gruppen gelegt.

Nicht nur dem schmalen Geldbeutel des Senators für Soziales, Jugend und Sport kommt diese Form der Selbstorganisation von Freizeit entgegen. Sie hilft, die Freizeitgestaltung nicht nur als Selbstzweck, sondern als sinnvoll über die Grenzen der Gruppen hinaus zu begreifen. Nicht nur, daß sich hier etliche nebenamtliche Gruppenleiter für eine fast symbolische Aufwandsentschädigung von fünf Mark pro Stunde bereitfinden, ihre Hobby-Kenntnisse weiterzuvermitteln: Es existieren rund hundert intakte Gruppen, die so nebenbei ihren Beitrag zum Funktionieren der Einrichtung leisten. Die schmucken Korblampen im gemütlichen Gemeinschaftsraum, in dem man abends noch beisammen sitzen und sich unterhalten kann, ohne gleich eine gepfefferte Wirtshausrechnung zu erhalten, sind selbstgeflochten. Der Festsaal, für jede Art von Feierlichkeiten kann er von jedermann für wenig Geld gemietet werden, ist geschmückt mit Wandmalereien der Künstlergruppe. Überall Spuren des Wirkens der Gruppen. Darauf kann man stolz sein.

Klar, daß da die Computergruppe nicht zurückstehen möchte. Seit Anfang April besteht sie erst, doch konnte sie den nicht unbedingt auf Computer eingeschworenen Besuchern des Zentrums ihre Nützlichkeit schon auf vielfältige Art beweisen. Als im Bürgerhaus ein Stadtteilfest stattfand, da entwarf man kurzerhand einen Fragebogen und wertete ihn selbst aus — auf dem Rechner selbstverständlich. So konnte man eine Menge von dem erfahren, was die Besucher vom Gemeinschaftszentrum halten.

Neue Ideen werden ausgetüftelt ...

Als der Senat seine riesigen Heizkosten für öffentliche Gebäude inspizierte, da konnte die Obervieländer Gruppe wieder einmal behilflich sein: mit einem Programm zur Berechnung der Fensterflächen, das von anderen Häusern sogar übernommen wurde. Auch die Kochgruppe nutzt inzwischen ein Rechnerprogramm — für den Zentraleinkauf von Lebensmitteln. Wer sich auf solche Art immer wieder nützlich machen kann, der ist natürlich gern gesehener Gast. Vielleicht ein guter Tip für so manchen Club, der noch ohne Domizil ist.



... in den Rechner eingegeben ...

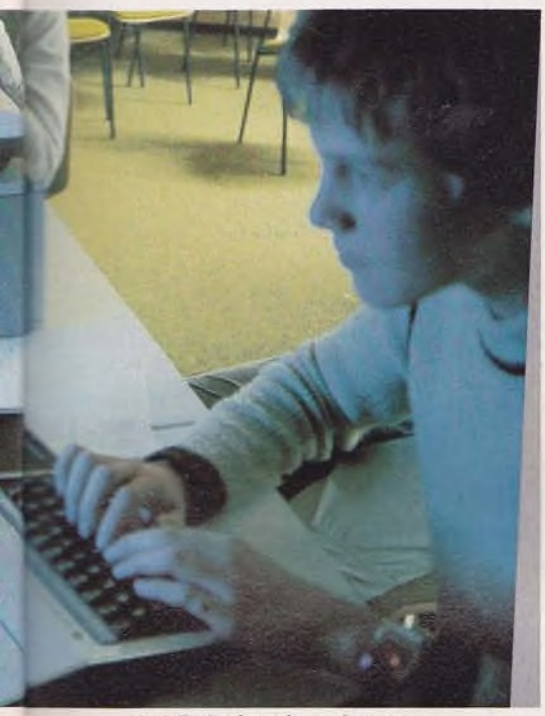
Gerhard Hollnagel, hauptamtlich im Bürgerhaus für die Technik angestellt, ist voller Begeisterung für seine Computerianer. Er selbst hat einen TI und war von Anfang an dabei. Er gehört zum 'harten Kern' der fünf Leute, die fast jeden Donnerstag anwesend sind. Ansonsten kommen mal mehr, mal weniger. Zwischen zehn und sechzig Jahre sind sie, auch Mädchen und Frauen darunter.

Einen eigenen Computer braucht man nicht zu besitzen. Das Bürgerhaus hat einen angeschafft — einen 99/4A, weil die meisten Gruppenmitglieder sich darauf auskennen. Dennoch sind auch Besitzer aller möglichen anderen Geräte herzlich willkommen. Schließlich wird man ja auch selbst dem Te-

Clubs

xas einmal Lebewohl sagen müssen, und es ist von Vorteil, dann schon jemanden zu kennen, der Erfahrungen auf einem anderen Rechner hat.

Einen besonderen Vorteil von der Gruppe haben aber momentan noch die TI-User. Schließlich betreibt man regen Programmtausch und kann eine Bibliothek von 400 Programmen vorweisen. Da dürfte für jeden Geschmack was dabeisein. Gegen einen freiwillig zu entrichtenden Obulus kann sich aus ihr jeder nach Herzenslust bedienen. Und mit dem so zusammengekommenen Geld



... bei Bedarf stetig verbessert ...

kann der Bestand leicht ergänzt werden, denn ein Großteil der Programme stammt aus Zeitschriften und wird, einmal eingetippt, schnell Allgemeinbesitz. Der Rest sind eigene Werke z.B. ein Editor für Schaltpläne, der sich im Rahmen der Aktivitäten des Hauses sicher noch von seiner nützlichen Seite zeigen kann.

Die gemeinsamen Abende dienen vor allem dem Informationsaustausch. Die erste halbe Stunde ist meist der Vorstellung von Neuigkeiten und Wissenswertem reserviert. Genug Zeit bleibt, sich einem Thema zuzuwenden, das, von einem zum anderen mal beschlossen, besonderes Interesse zu wecken vermag. Grafik auf dem TI beschäftigt die Gruppe zur Zeit, nachdem CP/T so einiges über die Fähigkeiten des auf diesem Gebiet verkannten Rechners ans Licht gebracht hat. Und die Gestaltung des Tages der offenen Tür im Bürgerhaus, auf dem man sich auch nach außen hin präsentieren möchte.

Ein paar Computerfreunde mehr mit vielen kreativen Ideen könnte man schon noch gebrauchen, zum Spaß und Nutzen.

Einen Computertag oder eine Ausstellung zum Thema Computer zu machen, das wäre ein mittelfristiges Ziel der Gruppe, um Bremen ein wenig aus dem Vorurteil der Computerprovinz herauszuhelfen. Denn um ein Vorurteil handelt es sich hier, wie wir feststellen konnten. Ist es doch beileibe nicht alltäglich, wie hier eine Gruppe von Computerbegeisterten ins kulturelle Leben eines ganzen Stadtteils verwoben ist.

Wer nicht so bald nach Bremen kommt, aber trotzdem Verbindung zu dieser so kontaktfreudigen Gruppe aufnehmen möchte, deren Mitglieder vielfach auch 'privat' gute Freunde geworden sind, der wende sich an die 1. Bremer TI-User Group im Gemeinschaftszentrum Obervieland, Alfred-Faust-Str. 4, 2800 Bremen 61, Tel.: 0421/82730.

Holger Neuhaus/Harald Uenzelmann

... bis sie schließlich den kritischen Blicken der User-Group standhalten.

In Kürze vorgestellt

Geschwindigkeit ist keine Hexerei, sagt man sich in Leverkusen, und ruft in Windeseile den **Sinclair-QL-User-Club** ins Leben. Soeben hat er seine Arbeit aufgenommen, die zunächst darin bestehen soll, Informationen über das schon vor Erscheinen legendäre Gerät zu verbreiten. Ein vierteljährlich erscheinendes Clubmagazin soll dazu beitragen. Weiteres Ziel soll die Bildung einer Programmbibliothek aus selbstgeschriebenen Programmen sein, die den Mitgliedern für wenig Geld zur Verfügung stehen soll.

Wer schon einen QL besitzt oder sich mit dem Gedanken trägt, einen anzuschaffen, und wem 20 Mark im Jahr nicht zuviel sind, der melde sich beim Sinclair QL User Club, Dirk Nitschke & Jörg Noa, Auf dem Bohnbüchel 1a, 5090 Leverkusen 3, 02173/42204.

Gute Kontakte zu heimischen Händlern besitzt der **Paradise Computer Club** Herborn. Das bedeutet für Mitglieder, die Atari oder Commodore-Spezis sind, günstige Einkaufmöglichkeiten in Sachen Hard- und Software. Doch nicht nur auf diese Weise will man Neulingen behilflich sein, auch auf den monatlichen Treffen werden reichlich Probleme gewälzt. Man erreicht den PCCH unter POB 1244, 6348 Herborn.

Bundesweit organisiert und mit reichlich ausländischen Kontakten versehen ist der **TI-Computer e.V.** Aachen. Einen Schwerpunkt legt der Verein, zu dessen Mitgliedern auch Professoren gehören, auf die ernsthafte Anwendung des 99/4A im mathematischen, physikalischen und astronomischen Bereich. Interessenten wenden sich an TI-Computer Club e.V., Hans-Böckler-Allee 155, 5100 Aachen, 0241-872205.

Alle Rekordjäger aufgepaßt! Anfang 1985 wird die vierte Saison der VCS-Bundesliga angepfiffen. Jeder, der auf Atari-Geräten original Atari-Software clubmäßig dattelt, kann sich zum Kräftemessen einfinden. Die Teilnahme ist erfreulicherweise kostenlos, ein Regelheft und Spielscheine können gegen die Schutzgebühr von zwei Mark beim **Atari-Mini-Club,** Blücherstraße 17, 6200 Wiesbaden bezogen werden.

Peek und Poke bein

Wer dem texanischen Instrument an die Maschine will, der erfährt hier stets eine Menge. Diesmal gibt's wichtige Informationen über Listschutz, die Communication Register Unit und was es mit den GROMs auf sich hat. Sie erfahren, wo der plauderhafte Speech-Synthesizer angesprochen wird und wo die vielgefragten Device-Service-Routinen liegen. Wir wünschen erfolgreiches Ausprobieren!

Mit PEEK und POKE bzw. CALL LOAD lassen sich natürlich auch Manipulationen in Extended BASIC Programmen durchführen. Ein kleiner Gag ist die bewußte Meldung eines Errors und der Abbruch des Programms:

```
CALL LOAD (-31965,3) ::
CALL LOAD (-31932,0)
```

Diese Zeile meldet einen Syntax Error — pikanterweise ohne Zeilennummer — und bricht das Programm ab. Wer nur die Programmausführung beenden will, läßt den ersten Call LOAD Befehl weg. Dabei wird mit ›8322 (dez. -31966) die Speicherstelle benutzt, die bei der Rückkehr von Assemblerprogrammen den Errorcode enthalten muß. Auf ›8344 (dez. -31932) steht während der Programmausführung ›FF. Wird dieses gelöscht, ist der TI wieder im Direktmodus.

Ein zusätzlicher Listschutz läßt sich durch eine Manipulation in der Zeilenliste durchführen. Die Zeiger auf dieser Zeilenliste stehen auf ›8332 (dez. -31950) für das Ende (letztes Byte) und ›8330 (dez. -31952) für den Anfang. Der Aufbau ist Zeilennummer, Zeiger auf die Zeile, Zeilennummer, Zeiger auf die Zeile usw. Mit

```
CALL PEEK (-31950,A,B)
```

suchen wir das Ende der Zeilenliste. Dann geben wir

```
CALL LOAD (A6+B-65536-3,127,255)
```

ein. Dabei wird aus A und B die Lage der ersten Zeilennummer errechnet. Die Werte 127,255 geben dieser Zeile die höchste beim TI 99/4A mögliche Zeilennummer. Bei einem LIST wird jetzt nur die erste Zeile im Programm aufgelistet. Das Listen wird immer abgebrochen, wenn die nächstfolgende Zeilennummer kleiner ist als die vorhergehende. Sehr wirksam ist dieser Schutz aber nicht, da durch RES die Aufhebung erfolgt. Wenn man allerdings eine zusätzliche Abfrage nach einer ganz bestimmten Zeilennummer, der man vorher eine möglichst unkonventionelle Nummer gegeben hat, im Programm zusätzlich versteckt, ist der Schutz nicht leicht zu brechen. Die Nummer der letzten Zeile ist recht einfach zu erhalten:

```
CALL PEEK (-31952,A,B) ::
PRINT A6+B
```

PEEK und POKE sind ja meist der erste Einstieg in die Welt der Maschinensprache. Für den täglichen Umgang allerdings ist eine genauere Kenntnis des TI 99/4A notwendig. Beim TI 99/4A liegt die bei anderen Computern vorhandene "Zero-Page" nicht in der gewohnten Form vor. In den letzten Ausgaben haben wir uns im Wesentlichen mit dem VDP-RAM beschäftigt. Es ist fast das einzige RAM in der Konsole und wird somit vom BASIC wie auch Extended BASIC intensiv genutzt. Um die Arbeitsweise des TI 99/4A besser zu verstehen, beschäftigen wir uns nun mit dem restlichen Aufbau, zumal dieser doch stark von dem üblichen abweicht.

Die Bremse beim TI

Der Prozessor im TI 99 ist der TMS 9900, eine vollwertige 16-Bit CPU, die aber zum Erweiterungssport 'gebremst' wurde. D.h.: hier sind nur acht Datenleitungen vorhanden. Der TMS 9900 besitzt 16 Adressenleitungen und kann so 64 KByte ROM oder RAM direkt adressieren. Eine Besonderheit des TMS 9900 ist die CRU (Communication Register Unit), die als eine Art serielle Schnittstelle mit zusätzlichem Adressenbereich betrachtet werden kann. Über diese Schnittstelle können an bestimmte Adressen Informationen (Bits) ausgegeben werden. Dabei sind diese Adressen aber nicht mit den Adressen von Speicherstellen zu verwechseln.

Bleiben wir aber zuerst bei dem 'normalen' 64 KByte Adressierungsbereich. Er wird in der TI 99/4A Konsole wie folgt genutzt:

›0000 bis ›1FFF:

Hier liegt das Betriebssystem, von Texas Instruments auch als Monitor bezeichnet. Dies enthält den Anfang der Powerup-Routine (läuft nach dem Einschalten des Gerätes ab), die Interrupt-Routinen zur Steuerung des automatischen Soundablaufes, der Spritebewegung und der Abfrage der Quit-Taste, diverse Routinen für die Rechnung mit Fließkommazahlen, die Cassetten-Ausgabe, einen Teil des TI-BASIC-Interpreters und den GPL-Interpreter.

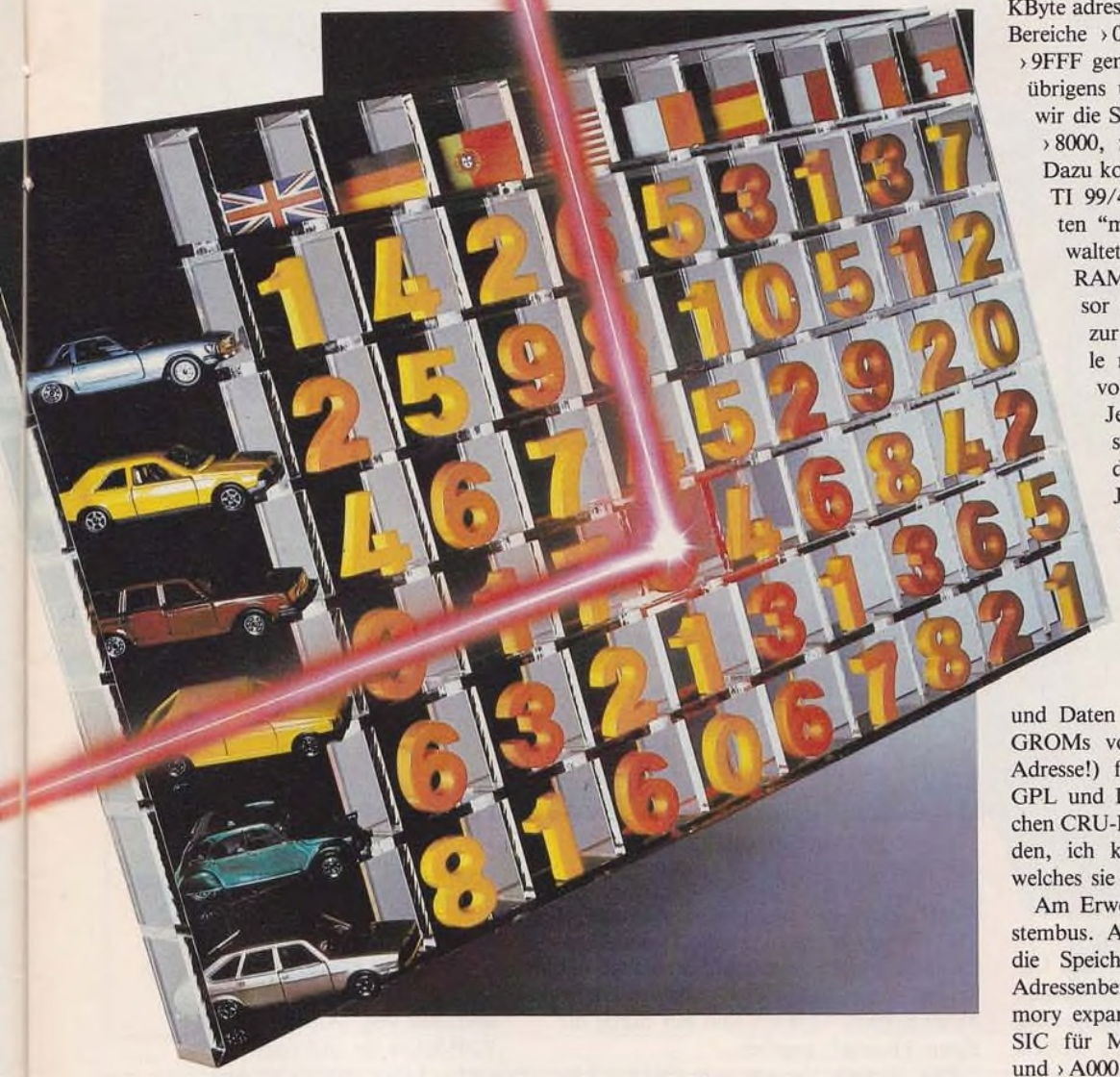
Was ist das nun schon wieder? GPL ist eine Programmiersprache, die Texas Instruments wohl nur deshalb einführte, um Softwareentwicklungen von dritter Seite für den TI 99/4A unmöglich zu machen, müssen doch Programme in dieser Sprache in ganz besonderen Bausteinen, den sogenannten GROMs abgelegt sein. Dieser Schutz-Versuch war dennoch vergeblich. Alle Routinen in diesem Adressenbereich von ›0000 bis ›1FFF sind natürlich in TMS 9900-Assembler geschrieben.

›8000 bis ›9FFF:

Hier befinden sich wichtige Systemadressen. Von ›8300 bis ›83FF befindet sich das einzige RAM in der Konsole, das direkt von der CPU adressiert werden kann. Entsprechend intensiv wird dieses RAM vom Betriebssystem genutzt. Es ist uns ja auch schon bei den Peeks und Pokes öfters begegnet. Auf ›8400 wird der Sound-Prozessor angesprochen und, sofern angeschlossen, auf ›9000 (Lesen) und ›9800 (Schreiben) der Speech-Synthesizer.

Weiter befinden sich in diesem Bereich die Adressen für zwei beim TI 99/4A wichtige Bauteile: den VDP und die GROMs. Beide arbeiten nach einem besonderen Prinzip. Sie haben einen internen Zähler, mit dessen Hilfe wieder weitere Speicheradressen angesprochen bzw. verwaltet werden. So verwaltet der VDP insgesamt 16 KByte RAM. Um auf eine bestimmte Speicherstelle dieses RAMs zuzugreifen, müssen wir also erst den Adressenzähler des VDP richtig setzen und können dann die Daten lesen. Um den Adressenzähler zu setzen, müssen wir auf eine bestimmte CPU-Adresse den Wert schreiben (Zwei Bytes nacheinander, da der Adressenzähler 16 Bits umfaßt). Dann können wir auf einer anderen CPU-Adresse Daten in das VDP-RAM schreiben oder auch aus dem VDP-RAM lesen. Die Adresse schreiben wir auf ›8C02, Daten lesen auf auf ›8800, und Daten schreiben wir auf ›8C00. Weiter können wir auf ›8802 das Statusregister des Video-Display-Prozessors lesen. Um nun den Zugriff auf das VDP-RAM zu erleichtern, erhöht sich der Adressenzähler des VDP bei jedem Zugriff auf Daten jeweils um eins, damit beim nächsten Zugriff nicht weiter die Adresse neu geschrieben werden muß, son-

n TI 99 4A



dern die Daten fortlaufend gelesen oder geschrieben werden können.

Prinzipiell genauso arbeitet ein GROM. Jeder dieser Bausteine besitzt einen 16 Bit-Zähler, mittels dessen das ROM in dem GROM adressiert wird. Auch dieser Zähler wird automatisch bei jedem Zugriff um eins erhöht. Auf ›8800 werden die Daten aus dem GROM gelesen und auf ›9C02 die Adressen geschrieben, wie beim VDP wieder 2 Bytes nacheinander. Nun kann bei einem GROM auch der Stand des Adressenzählers gelesen werden, auf der CPU-Adresse ›9802. Im TI 99/4A ist auch eine Adresse reserviert, um Daten in ein GRAM zu schreiben. Dies ist ›9C00. Nun existiert ein derartiger Baustein aber bis heute nicht. Mit einigem Hardwareaufwand dürfte sich so etwas aber nachbilden lassen, für uns aber momentan nicht interessant.

Unvollständig dekodiert: Der Bereich ›8000 bis ›9FFF

Der 16 Bit-Zähler im GROM ermöglicht die Adressierung von 64KByte (2^{16}) Speicher. Theoretisch ließe sich das in einem einzigen IC realisieren, aber im TI 99/4A umfaßt jedes GROM nur acht KByte, genau sind es sogar nur sechs Kbyte, die restlichen zwei KByte sind innerhalb jedes GROM's nicht dekodiert. In der Konsole eingebaut sind drei GROMs, auf den GROM-Adressenbereichen ›0000 bis ›17FF (Betriebssystem), ›2000 bis ›37FF und ›4000 bis ›57FF (Teil des BASICinterpreters).

Dieser etwas komplizierte Aufbau des TI 99/4A ist sicherlich auch Ursache für das Unverständnis, das diesem Heimcomputer von einigen Seiten entgegengebracht wird. Deshalb wollen wir noch einmal kurz wiederholen: Der TMS 9900 (die CPU) kann 64 KByte adressieren, wovon in der Konsole die Bereiche ›0000 bis ›1FFF und ›8000 bis ›9FFF genutzt sind. Der letzte Bereich ist übrigens unvollständig dekodiert, so daß wir die Speicherstelle auf ›8300 auch auf ›8000, ›8100 und ›8200 wiederfinden. Dazu kommen weitere Speicherstellen im TI 99/4A, die nach dem geschilderten "memory mapped"-Verfahren verwaltet werden. So stehen 16KByte RAM über den Video-Display-Processor und insgesamt 64KByte GROM zur Verfügung, wobei in der Konsole nur insgesamt 18KByte GROM vorhanden sind.

Jetzt ist die Konsole nahezu vollständig abgeschlossen. Bleibt nur die Frage nach der Tastatur, den Joystick-Ports und dem Cassettensetten-Port. Diese werden über die erwähnte CRU abgefragt bzw. betätigt.

Das nächste ist nun der Modulport. An ihm steht der CPU-Adressenbereich von ›6000 bis ›7FFF für Assemblerprogramme und Daten zur Verfügung. Dazu kommen GROMs von ›6000 bis ›FFFF (GROM-Adresse!) für Programme in der Sprache GPL und Daten. Auch sind die erforderlichen CRU-Leitungen am Modulport vorhanden, ich kenne aber bisher kein Modul, welches sie ausnützt.

Am Erweiterungsport liegt der ganze Systembus. Angeschlossen wird üblicherweise die Speichererweiterung. Sie belegt den Adressenbereich ›2000 bis ›3FFF (low memory expansion, wird unter Extended BASIC für Maschinenprogramme verwendet) und ›A000 bis ›FFFF. Dann wird es schon wieder etwas komplizierter. Auf ›4000 bis ›5FFF liegen die Device Service Routinen (DRS), die Betriebssysteme für die Peripherie wie die RS232-Schnittstelle und den Diskettenkontroller. Nun können diese natürlich nicht alle parallel liegen, sondern werden über CRU-Adressen abwechselnd eingeschaltet. In der Praxis sieht das so aus, daß bei dem Aufruf (z.B. durch OPEN) die CRU-Adressen von ›1000 bis ›1FFF in Schritten von ›100 abgesucht werden, ob ein DSR-ROM vorhanden ist (Kennbyte ›AA auf ›4000). Wenn ja, wird die DSR-Liste (Zeiger darauf steht auf ›4008) nach dem gewünschten Namen abgesucht. Ist der Name gefunden, so steht vor dem Namen die Einsprungadresse der Routine, die dann ausgeführt wird.

Heiner Martin

Objektcode-Futter

In Heft 1/85 stellten wir Ihnen den Objektcode-Editor vor, eine tolle Möglichkeit, TI-Maschinensprache ohne Assemblierung zu realisieren. Damit Sie in Übung bleiben, bringen wir hier zwei weitere Anwendungen.

Scroll und Spirale

Um die Gestaltung von Spielprogrammen etwas interessanter zu machen, wurden die folgenden Routinen geschrieben. Sie werden aufgerufen durch:
CALL LINK("SCROLL",mode,string)
wobei mode=1 SCROLL UP, mode=2 SCROLL DOWN, mode=3 SCROLL LEFT, mode=4 SCROLL RIGHT bewirkt. Unter "string" kann maximal ein 32 Zeichen String stehen, der in die freie Reihe/Spalte geschrieben wird.

Es besteht also die Möglichkeit, sich einen überdimensionalen Schirm in einem Stringarray zu definieren und z.B. durch Joysticks diesen Schirm dann nach rechts, links, oben, unten zu durchwandern. In Verbindung mit Sprites ergeben sich noch mehr ungeahnte Möglichkeiten. Grenzen sind nur durch die eigene Phantasie gegeben...

Eine reizvolle Variante zum CALL CLEAR des ExBASIC stellt die Spirale dar. Beginnend in der Mitte wird der Bildschirm spiralförmig gelöscht. Der für diese Routine nötige Aufruf ist:
CALL LINK("SPIRAL")
Auf diese Art und Weise bekommen Sie einen 'Loadrunner'-mäßigen Screenwechsel.

VDP Routinen

Das Lesen bzw. Schreiben des VDP-RAMs ist in ExBASIC leider nicht möglich. Wenn Sie Bitmap, Mulitcolor oder Text ausprobieren wollen, benötigen Sie eine Hilfsroutine. Dies war die Motivation für das Erstellen des folgenden Programms.

Das Programm stellt folgende Routinen zur Verfügung:

- ▷ CALL LINK ("POKEV",Anz, Adresse, Array)
  Schreiben von ‹Anz› Bytes ins VDP-RAM ab ‹Adresse›
- ▷ CALL LINK ("PEEKV",Anz, Adresse, Array)
  Lesen von ‹Anz› Bytes ab ‹Adresse›
- ▷ CALL LINK ("VDPR",Wert,Reg)
  Schreiben von ‹Wert› in VDP-Register ‹Reg›

Für POKEV und PEEKV befinden sich die Werte jeweils in einem Array, das max. 767 Elemente enthalten darf und mit einem entsprechenden DIM Befehl im Basic Programm deklariert werden muß.

Ehe wir es vergessen: Zum Abspeichern der Daten brauchen Sie ein Diskettenlaufwerk. Vielleicht findet sich ja einer unter Ihnen, der eine Cassettenroutine zum Abspeichern der Datensätze schreibt. Wir wünschen auf jeden Fall viel Vergnügen mit unseren Nützlichkeiten in Maschine!

H. Amshove

SCROLL: Ihr Wunsch nach Textblock-Bewegung wird mit dieser Routine endlich Wirklichkeit. Kontrolliertes Scrolling in jede Richtung ist nun kein Problem mehr.

```
00492SCROLL   A0000A0
300A0302A0324A0344B0
300B0000B0420B200CB0
4207F26CF        0001
A034EB2018B12B8B0300
B0002BC4A0B834AB8ED2
B1104B0200B1E00B0420
7F2CAF           0002
A0364B2034B8ED2B1504
B0200B1E00B0420B2034
B045BB02E0C0324B0200
7F2FAF           0003
A037AB0000B0201B0001
B0202C0300B06A0C0344
B0005B0000B0200B0000
7F361F           0004
A0390B0201B0002B0202
C0302B0203B2000BC483
B0420B2014B0200B0000
7F35FF           0005
A03A6B0201C0000B0202
B0300B0420B202CBC1E0
C0300B0287B0002B151D
7F315F           0006
A03BCB0204B0017B0205
B0020B0208B0001B0202
B0001B0287B0001B1609
7F341F           0007
A03D2B0209C0000B0206
B0000B020AC0020B0201
C02E0B1032B0209C02E0
7F32BF           0008
A03E8B0206BFFC0B020A
C02C0B0201C0000B1029
B0204B0018BD0A0C0302
7F2D8F           0009
A03FEB0982B0282B0018
B1104B0202B1800BD802
C0302B0205B001FB0202
7F303F           0010
A0414B0020B0287B0003
B160BB0206B0001B0208
B0001B0209C0000B020A
7F347F           0011
A042AC0001B0201C001F
B100BB0206B003EB0585
B0208BFFFFB0209C001F
7F2B4F           0012
A0440B020AC001EB0201
C0000BC1C5BD65ABA288
BA248B0607B16FBBA286
7F281F           0013
A0456BA246B0604B16F6
BD1E0C0302B0987B0200
C0303BD1B0B0226B6000
7F2D1F           0014
A046CBD446BA042B0607
B16F9B0200B0000B0201
C0000B0202B0300B0420
7F316F           0015
A0482B2024B04C0BD800
B837CB02E0B83E0B0460
B00707F602F      0016
50374SCROLL7FCF7F
                 0017
:        99/4 AS 0018
```

SPIRAL: Nun können Sie schöne Spielausblendungen vornehmen.

```
000F4SPIRAL   A0000A0
020B8000B020AB03E8B0
60AB16FFB045BBC1CBB0
2B87F1B6F
A0030B0004B1102B0420
B2020B06A0C0022B0580
B0605B16F6B0457BC1CB
7F2FDF
A0046B0420B2020B028B
B001CB1502B0420B2020
B06A0C0022B0600B0605
7F339F
A005CB16F6B0457BC1CB
B0420B2020B06A0C0022
B0220B0020B0606B16F8
7F2D1F
A0072B0457BC1CBB0209
B0020B0420B2020B06A0
C0022B6009B0606B16F9
7F2ECF
A0088B0457BC148B6142
B05858C184B6183B0586
B0286B0017B1102B0206
7F2EAF
A009EB0017B045BB02E0
C0000B0200B017080201
B8000B0202B0010B0208
7F335F
A00B4B0010B0203B000C
B0204B000CB06A0C008A
B06A0C002CB06A0C0060
7F2EEF
A00CAB0588B0288B0020
B130AB0584B06A0C008A
B06A0C0044B06A0C0074
7F2CBF
A00E0B0602B0603B1QEC
B04C0BD800B837CB02E0
B83E0B0460B00707F3C5
F
500A2SPIRAL7FCF6F
:        99/4 AS
```

VDP-ROUTINEN: Schneller Zugriff, diesmal in anderer Form.

```
000EEVDPAC    A0000A0
020A0022A0024A0026B0
000B00FFB0301B4001B0
0007F289F        0001
A0030B0200B1C00B0420
B2034BD820C002EB837C
B02E0B83E0B0460B0046
7F2E4F           0002
A0046B0300B0000B0420
B200CB0020B0420B2008
B0300B0002B045BB0300
7F355F           0003
A005CB0000B0420B200C
B0420B2018B12B8B0300
B0002BC4A0B834AB045B
7F2E9F           0004
A0072B0460C0038B0200
B0000B0201B0001B0202
C0024B06A0C005ABC0D2
7F32BF           0005
A0088B0283C002AB1102
B0460C0030B0201B0002
B0202C0022B06A0C005A
7F333F           0006
A009EBC0D2B0283C002C
B1102B0460C0030B045B
B02E0C0000B06A0C0072
7F2E2F           0007
A00B4BC1A0C0022B0202
C0020BC160C0024B0200
B0001B0201B0003B06A0
7F32AF           0008
A00CAC005ABC0D2B0283
C0028B1102B0460C0030
BC060C0021B0200B0006
7F2FFF           0009
A00E0B0420B2020B0586
B0605B16EBB0460C0038
7F739F           0010
500ACPOKEV 7FD0BF 11
:        99/4 AS 0012
```


Der TI 99/4A weiter auf dem Vormarsch

Viele interessante Neuheiten – vieles sofort lieferbar.

Immer, wenn Sie Ihren TI 99 / 4 A leistungsfähiger und vielseitiger machen wollen – greifen Sie zu. Bauen Sie Ihre Damast-Datenbank aus.

NEU aus USA

Disketten-Laufwerk
Intern, doppelseitig, Slimline mit halbem Stromverbrauch. Zum problemlosen Einbau zweier Laufwerke in die Peripherie-Box. Komplett mit Kabelsatz. Je Laufwerk **DM 798,–**

Externe Kombination
Verzichten Sie auf riesige Boxen. Ein Minigehäuse, ganze 13,7 x 14 x 6,3 cm klein, bietet Ihnen wahlweise:
- 3 Schnittstellen extern (1 x Parallel und 2 x V.24), Bus nicht durchgeführt, nachträglich komplett aufrüstbar **DM 498,–**

Modul-Expander
- 3fach, zum Einschieben in den Schacht, aus den USA, sofort lieferbar **DM 128,–**
- 8fach, „Made in Germany", binnen 2 Wochen lieferbar **DM 198,–**

Grafik-Tablett Super-Sketch
Sie zeichnen das Bild auf das Tablett, Super-Sketch überträgt die Grafik sofort und unmittelbar auf den Bildschirm. Binnen 2 Wochen lieferbar. **DM 248,–**

In Kürze lieferbar:
RGB-Modulator, AD/DA-Wandler

Programmiersprachen
- Extended Basic Modul Original TI **DM 278,–**
- Mini Memory 4K CMOS-RAM-Modul **DM 278,–**
- Editor Assembler Paket **DM 178,–**
- TEX-FORTH-Diskette für das Editor Assembler Modul. Kann 64 Zeichen auf dem Bildschirm darstellen. Mit englischem Handbuch. Sofort lieferbar. **DM 98,–**
- Pascal-System P-Code-Karte mit Disketten und Manual **DM 998,–**

Wichtige Software-News Module aus USA
Als größter deutscher apesoft-Händler liefern wir – auch Neuheiten – ständig ab Lager.

Software Extended Basic / 32 K RAM
Grafic Master, Diskette **DM 99,–**
Vollintegrierte Hardcopy-Routine dazu **DM 30,–**
3-D-World, Diskette **DM 99,–**
Vollintegrierte Hardcopy-Routine dazu **DM 30,–**

Super Sonderangebot:
DISKY 5,25" 1 D, 40 Spuren einzeln geprüft. **DM 5,50**

Neu! DISKY 5,25" Two eye Die Wendediskette. **DM 6,90**

Spielmodule neu aus USA
zwischen **DM 89,–** und **DM 98,–**

Basic Computer. Neu! Bitte nachfragen!
Fordern Sie bitte unbedingt unsere ausführliche Preisliste an.

Das Computer-Hüsli
Münchner Straße 48
8025 Unterhaching
Telefon: 0 89 / 61 90 48

Tontauben-Schießen

Der TI auf dem Olympia-Schießstand

Für den TI 99/4A ein Programm zu schreiben, in dem Sprites nicht nur aus dekorativen Gründen zum Einsatz kommen, ist eine ganz schön komplizierte Angelegenheit. Das Problem liegt, wie alle TI-User wissen, bei der viel zu langsamen Positions-Abfrage der Sprites durch den BASIC-Interpreter.

Dennoch ist es möglich, wie uns dieses Programm von Peter Hlawna beweist. Unter Ausnutzung fast aller CALL-Routinen, die das TI-BASIC anbietet, hat Peter uns ein außerordentlich interessantes Programm vorgelegt. Alle Achtung!

Wie es sich für eine Tontaube gehört, fliegt sie vom Schützen weg. Und das natürlich in einer vernünftigen Kurve, unter Berücksichtigung der perspektivischen Gegebenheiten, nach allen Regeln der Kunst.

Gesteuert wird das Zielkreuz, in das man die Tontaube bekommen muß, um sie abzuschießen, mit dem Joystick 1. Fünfzig Tontauben gilt es, vom Himmel zu holen. Wem das zu viele sind, der ändere einfach in Zeile 910 die IF TAU = 51... Abfrage um. TAU ist die Variable, die die Zahl der Tontauben führt. Setzt man in dieser Abfrage also zum Beispiel 31 ein, so braucht man nur noch 30 Tontauben abzuschießen.

Obwohl das Programm für das, was es leistet, extrem kurz ist (genau 115 Programm-Zeilen), gilt es, beim Eintippen größte Sorgfalt aufzuwenden. Ein Tippfehler in den DATAs oder den DISPLAYs, und schon verrutscht die schöne Grafik.

Zum Listing

Zeile 100: Initialisierung

230—710: Die DATAs für die einzelnen Elemente der Grafik. Zeile 710 liest die DATAs ein und definiert die nötigen Character der Ansehnlichkeit halber um.

750—930: Aufbau der Bildschirmgrafik. Hier gilt es, mit großer Sorgfalt zu arbeiten, um keine Fehler in die wirklich sehr schöne Grafik zu bekommen. In den DISPLAY-Befehlen handelt es sich übrigens immer um 'O's und nicht um Nullen.

970—1130: Das Spiel. Gespielt wird mit Joystick 1. Die Tontaube wird mit dem Feuerknopf gestartet, danach wird mit ihm ganz normal geschossen. Aber Vorsicht, daß kein Vogel getroffen wird!

1170—1240: Endstatistik. Aus den Treffer-Punkten wird die prozentuale Abschuß-Quote ermittelt. Ein Druck auf den Feuerknopf startet das Spiel erneut.

Variable:

A$;	READ-Variable
I;	FOR/NEXT-Variable
VO‹›;	Funktion

Tontauben-Variablen:

TAU;	Tontaubenzähler
COUNT;	Status
C;	Status-Hilfsvariable
XM,YM;	Geschwindigkeit und Richtung
PA;	Form

Vogel-Variablen:

V;	Zähler für Treffer
XV,YV;	Position

Spieler-Variable:

P;	Punkte
PRO;	Prozent
X, Y;	Joystick
K, S;	Feuerknopf
A, B;	Trefferabfrage
TOL;	Treffertoleranz

SPRITE-Tabelle:

1	Zielkreuz
4	getroffener Vogel
10	Tontaube
15—19	Vögel

```
100 CALL CHARSET :: CALL CLEAR :: TAU,V,P=0 ::
    RANDOMIZE :: DEF VO(X)=INT((X+260)/20)
110 ! *{CHR$(126)}*{CHR$(126)}*{CHR$(126)}
    *{CHR$(126)}*{2*CHR$(126)}*{CHR$(126)}
    *{CHR$(126)}*{CHR$(126)}*{CHR$(126)}*
120 ! *TONTAUBENSCHIESSEN*
130 ! {20*(*)}
140 ! * PROGRAMMIERT VON *
150 ! *   PETER HLAWNA   *
160 ! *    JUNI 1984     *
170 ! *{CHR$(126)}*{CHR$(126)}*{CHR$(126)}
    *{CHR$(126)}*{2*CHR$(126)}*{CHR$(126)}
    *{CHR$(126)}*{CHR$(126)}*{CHR$(126)}*
180 !
190 ! ZEICHENDATAS
200 !
210 ! GESCHUETZ
220 !
230 DATA 1E1E1E1E3C3C3C3C,FFFF7C7CECECECEC,FFFF,
    0101010103030303
240 DATA ECECECE{9*(C)},070707070F0F0F0F,
    8C8C8C8C0C0C0C0C,0C0C0C0C0C0C0C0C,
    FFFF0C0C0C0C0C0C
250 DATA ,,01071921274948FC,4849272119070100,
    00C03008C824247E,2424C80830C00000
260 !
270 ! ZAHLEN
280 !
290 DATA 7CFECEDEF6E6FE7C,1838787818181818,
    7CFE0E0E3C70FEFE,7CFE063C3C06FE7C
```

```
300 DATA C0C0D8D8FEFE1818,FCFEC0FCFE06FEFC,
    7CFEC0FCFEC6FE7C,FEFE0E0E1C3870E0
310 DATA 7CFEC67C7CC6FE7C,7CFEC6FE7E06FE7C,,
320 !
330 ! VOEGEL
340 !
350 DATA 0000E0381E0F3F27,,{6*(0)}60D0FEE0C0,
360 DATA 1149250060002040,00D2221A28,
    1024480106000007,000609A4AA24
370 DATA 0000301C0F1F1F0F,0101,0070A8F8D8F0E0C0,
    0080
380 !
390 ! GEBIRGE
400 !
410 DATA {6*(0)}1D3F3F7FFF,01030787C{7*(F)}
    ,C0E0F0F9FB{6*(F)},{6*(0)}80C0F0F8FF
420 DATA 03070F0F1F3F3FFF,8080C0F0F8FEFFFF,
    000004CE{8*(F)},{16*(F)}
430 !
440 ! BACH
450 !
460 DATA {7*(0)}609304080,{8*(0)}
    E31C0000,011A24C0,{8*(0)}3AC5
470 DATA 000060908C03,{10*(0)}
    E01C03,80402619,00186681
480 !
490 ! SEE
500 !
510 DATA {7*(0)}30F1F3FFF,00003C{10*(F)}
    ,000{13*(F)},{16*(F)}
520 DATA F1FFFFE7C3E3FFFF,A0F1{12*(F)},F1{14*(F)}
    ,E0FCFEFFFEFCE0C0
530 DATA FF3F07,{6*(F)}E701,{10*(F)}
    3F1F03,{13*(F)}0C0
540 DATA {12*(F)}DEC0,{8*(F)}C780,FFF0C080,
550 !
560 ! WALD
570 !
580 DATA 00040A2052051028,40A40A2050022550,
    40A40A0000020500,40A40A
590 DATA 40A40A0000020500,,,
600 !
610 ! TONTAUBEN
620 !
630 DATA {7*(0)}30C336FDF,DF6F3F0F03,{6*(0)}
    C030CCFEFF,FFFEFCF0C0
640 DATA {9*(0)}30C336F,6F3F0F03,{8*(0)}
    C030DCFE,FEFCF0C0
650 DATA {11*(0)}30C33,3F0F03,{10*(0)}
    C030FC,FCF0C0
660 DATA {11*(0)}30C1B,1F0F03,{10*(0)}
    C030F8,F8F0C0
670 DATA {11*(0)}1070F,0701,{10*(0)}80E0F0,E080
680 DATA {13*(0)}107,01,{12*(0)}80E0,80
690 DATA {13*(0)}103,01,{12*(0)}80C0,80
700 DATA {15*(0)}1,,{14*(0)}80,
710 RESTORE :: FOR I=33 TO 143 :: READ A$ ::
     CALL CHAR(I,A$):: NEXT I
720 !
730 ! BILDSCHIRMAUFBAU
740 !
750 CALL CLEAR :: CALL SCREEN(2)::
```


Unser Überlebenskonzept:

O & MD Hamburg

Gewaltfreie Aktionen in aller Welt zur Durchsetzung folgender Ziele:
- Beendigung aller Atombombentests
- Stop der Versenkung von Chemie- und Atommüll im Meer
- Erhaltung der Robben, Wale und Delphine
- Stop dem Sauren Regen und der weiteren Vergiftung unserer Umwelt durch Schadstoffe und Abgase

Greenpeace dankt für den kostenlosen Abdruck dieser Anzeige.

Ich will das Greenpeace-Überlebenskonzept für unsere Umwelt kennenlernen. Senden Sie mir ausführliches Informationsmaterial.

Name/Vorname

Straße/Nummer

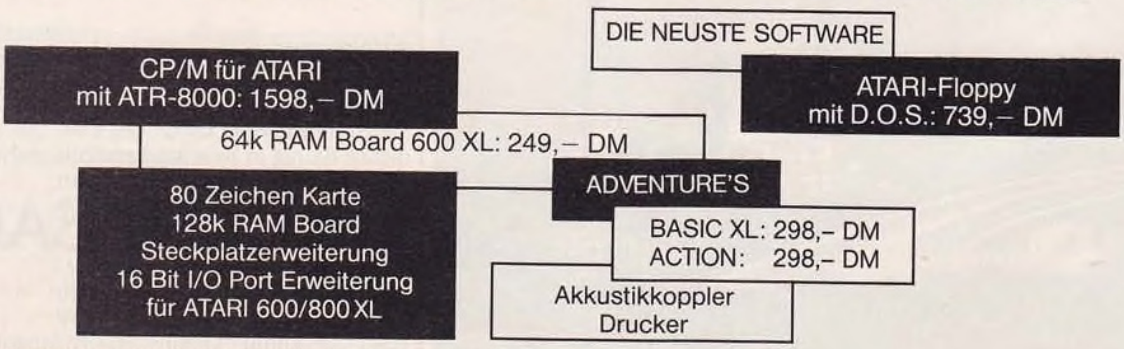
PLZ/Wohnort A 242

Sie zu informieren kostet uns DM 2,40. Schicken Sie diesen Betrag in Briefmarken (falls mehr, bitte Scheck) zusammen mit diesem Coupon an:

GREENPEACE
Greenpeace - Informationsdienst
Hohe Brücke 1 - Haus der Seefahrt -
Telefon: 040/37 33 44 oder 37 33 59
Bankverbindung: Vereins- und Westbank, Hamburg
Konto-Nr. 1/471300, BLZ 200 300 00

```
     CALL MAGNIFY(3)
760 CALL HCHAR(1,1,112,224)::
     CALL VCHAR(8,1,112,2)::
     CALL VCHAR(8,2,112,3)::
     CALL VCHAR(8,31,112)::
     CALL VCHAR(8,32,112,2)
770 CALL HCHAR(10,1,77):: CALL HCHAR(11,2,74)::
     CALL HCHAR(9,31,75):: CALL HCHAR(10,32,77)
780 CALL VCHAR(11,1,79,3)::
     CALL VCHAR(12,2,79,2)::
     CALL VCHAR(10,31,79,4)::
     CALL VCHAR(11,32,79,3)
790 DISPLAY AT(8,1):"{11*(p)}HNNK{10*(p)}
    HNpppHKHIMpppLOOOOM{8*(p)}
    LOOJpLOOOOOMHI{6*(0)}JNKppppLOOOOI{18*(0)}
    MppLOOOOO"
800 DISPLAY AT(12,1):"{20*(0)}JI{6*(0)}"
810 CALL HCHAR(13,3,79,28)::
     CALL HCHAR(16,1,80):: CALL HCHAR(16,2,81)
820 CALL VCHAR(14,31,105,7)::
     CALL VCHAR(14,32,105,7)
830 DISPLAY AT(14,1):"{23*(SPACE)}
    hiiii PSQTSQU   XYZ[[\]^[_QUiiiiiR{7*(SPACE)}
    VWR{CHR$(96)}ab[[cdef   jiiii{25*(SPACE)}k1i"
840 DISPLAY AT(18,1):"{27*(SPACE)}
    i  ! !{22*(SPACE)}j"
850 DISPLAY AT(20,1):"  "&CHR$(34)&"#"&CHR$(34):
    " $%$%":" &'&'":"x!(!(":
    " "&CHR$(34)&")"&CHR$(34)&"("
860 CALL COLOR(1,7,4,2,7,4,3,5,4,4,5,4,6,2,8,7,5,
    4)
870 CALL COLOR(8,5,4,9,5,4,10,13,4,11,8,8,12,4,4)
880 FOR I=15 TO 19 :: CALL
     SPRITE(#I,60,13,I*20-250,1,0,INT(RND*8)+5)::
     NEXT I
890 DISPLAY AT(24,28):"0"
900 CALL SPRITE(#1,44,12,92,128)
910 CALL SPRITE(#10,112,14,175,30):: PA=112 ::
     C,COUNT=1 :: TOL=10 :: TAU=TAU+1 ::
     IF TAU=51 THEN 1170
920 DISPLAY AT(24,10)SIZE(2):USING "##":TAU
930 XM=INT(RND*6)+3 :: YM=INT(RND*13)+14+V/2
940 !
950 ! SPIELBEGINN
960 !
970 FOR I=1 TO 30 :: CALL JOYST(1,X,Y)::
     CALL MOTION(#1,-6*Y,6*X):: NEXT I
980 CALL MOTION(#10,-YM,XM)::
     FOR I=1 TO 30 STEP 5 ::
      CALL SOUND(-100,330-4*I,I)::
      CALL SOUND(-100,230-4*I,I):: NEXT I
 990 CALL JOYST(1,X,Y):: CALL MOTION(#1,-6*Y,6*X)
     :: CALL KEY(1,K,S):: IF K=18 THEN 1030
1000 YM=YM*.75 :: CALL MOTION(#10,-YM+2,XM)::
      COUNT=COUNT-1 :: IF COUNT>0 THEN 990
1010 PA=PA+4 :: IF PA>140
     THEN CALL DELSPRITE(#10)::
      GOTO 910 ELSE CALL PATTERN(#10,PA)
1020 TOL=TOL-1.5 :: C=C*1.4 :: COUNT=C :: GOTO 990
1030 COUNT=COUNT-1 :: CALL POSITION(#1,Y,X)::
      CALL COINC(#1,#10,TOL,A)::
      CALL COINC(#1,#VO(Y),8,B)
1040 IF B=-1 THEN CALL MOTION(#VO(Y),
     0,0)ELSE IF A=-1 THEN CALL MOTION(#10,0,0)
1050 FOR I=1 TO 24 STEP 3 ::
      CALL SOUND(-10,-6,I):: NEXT I
1060 IF A=0 AND B=0 THEN 1000
1070 IF B=-1 THEN 1090
1080 CALL DELSPRITE(#10):: FOR I=1 TO 15 STEP 3 ::
      CALL SOUND(-100,-5,I):: NEXT I :: P=P+30 ::
      DISPLAY AT(24,24):USING "#####":
     P :: GOTO 910
1090 CALL PATTERN(#VO(Y),68)::
      CALL COLOR(#VO(Y),10)::
      CALL POSITION(#VO(Y),YV,XV)::
      CALL SPRITE(#4,64,3,YV,XV)
1100 FOR I=1 TO 15 STEP 3 ::
      CALL SOUND(-100,-7,I,1000,1):: NEXT I
1110 CALL DELSPRITE(#4):: CALL MOTION(#VO(Y),
     25,0):: FOR I=330 TO 110 STEP -20 ::
      CALL SOUND(-100,I,I/20):: NEXT I
1120 CALL DELSPRITE(#VO(Y)):: V=V+1 :: P=P-200 ::
      IF P<0 THEN P=0
1130 DISPLAY AT(24,24):USING "#####":P :: GOTO 910
1140 !
1150 ! SPIELENDE
1160 !
1170 CALL CLEAR :: CALL DELSPRITE(ALL)::
      CALL CHARSET
1180 FOR I=1 TO 8 :: CALL COLOR(I,16,2):: NEXT I
1190 DISPLAY AT(10,5):"G A M E  O V E R"
1200 PRO=INT(P/15)-(P/15-INT(P/15)>=.5)
1210 DISPLAY AT(16,5):"SCORE: ";P
1220 DISPLAY AT(18,5):"DAS ENSPRICHT";PRO;"%."
1230 DISPLAY AT(24,1):"PRESS BUTTON TO START
      AGAIN"
1240 CALL KEY(1,K,S):: IF S=0 THEN 1240 ELSE 100
```


HAASE-Computersysteme – Ihr ATARI-Fachmann:



Bestellungen und Informationen bei:
HAASE-Computersysteme, Wiedfeldtstraße 11, D-4300 Essen 1, Tel. (02 01) 42 25 75

See-Saw

Hersteller: Quicksilva
System: Commodore 64
Programm: Cassette

Vorsicht, Steinschlag!

Ein boshafter Burgherr hält die Brüder des Hauptdarstellers in finsteren Burgverliesen gefangen. Nur der Spielheld kann sie retten, indem er mittels einer Wippe die Burgmauern überspringt und die Kerkertüren öffnet. Doch daran wird er durch den Burgherrn, der schwere Steine auf ihn wirft und seine Helfer, eine Meute grüner, gehässiger Monster, gehindert.

Aufgabe des Spielers, der den Befreier mit dem Joystick steuert, ist es, diesen so geschickt auf der Wippe zu plazieren, daß ihn die herunterfallenden Steine heraufschleudern. Vorsicht: Die grünen Monster grabschen alles an, was an ihnen vorbeifliegt. Ist dies ein Stein, dann reißt er sie mit in die Tiefe, handelt es sich dabei jedoch um den Retter, verliert dieser eines seiner drei Leben.

Der Spieler muß zunächst die Reihen seiner Feinde derart lichten, daß er an der entstandenen Lücke vorbeispringen kann, um auf die Burgzinne zu gelangen. Dabei kann es von Vorteil sein, die auf der Wippe gelandeten Steine an eine andere Stelle zu schieben, um die Flugbahn des hinaufkatapultierten Helden zu beeinflussen. Hierbei ist jedoch zu beachten, daß mehr als zwei Steine auf einer Seite der Wippe nicht bewegt werden können und mehr als fünf Steine auf der Wippe diese unbeweglich machen. Der Spieler kann das Ziel nicht erreichen. Erschwerend kommt außerdem hinzu, daß nicht vorhergesehen werden kann, welchen Stein der böse Burgherr als nächsten nach unten werfen wird. Glück gehört zum Gewinn.

Hat der Spieler die erste Spielstufe gemeistert, kann es ihm bei den höheren Schwierigkeitsgraden passieren, daß Steine mit doppeltem Gewicht heruntergeworfen werden. Man erkennt diese an ihrer schwarzen Rückseite leider erst sehr spät vor dem Aufprall. Manchmal zu spät...

"See-Saw" ist ein Spiel mit witzigem Handlungsablauf, das besonders jüngere Computerfreunde ansprechen wird. Den Ton sollte man am besten abschalten, da der unangenehme Klang nach kurzer Zeit an die Nerven geht. Doch was soll's? Man kommt auch ohne sie per Wippe gut in Schwung.

Björn Schwarz

	Note		Note
Grafik:	❷	**Action:**	❸
Sound:	❺	**Spielwitz:**	❷
Gesamtergebnis:		**Spielidee:**	❸

Qbert

Hersteller: Parker Brothers
System: TI
Programm: Modul

Der richtige Riecher

Wer Qbert ist, sollte mittlerweile hinreichend bekannt sein. Es handelt sich um den gnomenhaften Kerl mit der Elefantennase, sprich Rüssel mitten im Gesicht. Wie sein großer Bruder in den Spielhallen lebt auch unser Held auf einer Würfelpyramide. Sein einziger Lebensinhalt scheint darin zu bestehen, auf seiner Pyramide umherzuspringen, um die einzelnen Würfelflächen durch seine Berührung in neuer Farbe erstrahlen zu lassen. Das wäre einfach, wenn es nicht noch ein paar Gesellen gäbe, die unserem Helden ans Leder wollen. Da gibt es rote und violette Bälle, die Qbert ins Nirwana schießen möchten. Genauso gehässig ist Coily, eine Schlange, die unseren Qbert mit grausamen Geräuschen verfolgt. Slick and Sam sind von besonderer Güte. Sie färben die von Qbert schon berührten Würfelflächen in ihre alte Farbe zurück. Dann muß sie der kleine Gnom nocheinmal bemühen.

Aus diesem Schlamassel kann sich Qbert retten, indem er einen grünen Ball fängt. Alle Gegner sind dann für einen kurzen Moment gelähmt. Er kann in Ruhe weiterarbeiten, und es gibt Bonuspunkte. Nicht nur durch Feindberührung sondern auch durch ungeschicktes Hantieren mit dem Joystick kann Qbert eines seiner Leben verlieren. Nämlich wenn er von seiner geliebten Pyramide im eleganten Tarzansprung hinunter ins Nichts hechtet. Das sieht zwar toll aus, beendet das Spielgeschehen aber auch recht schnell.

Dummerweise sind die Joystickabfragen vom Hersteller ungeschickt gewählt. Um vernünftig spielen zu können, müssen Sie den Stab der Freude diagonal halten. Eine ungewohnte Handhabung, die zu Anfang große Schwierigkeiten bereitet. Das ist aber auch der einzige Nachteil dieses Spiels. Wer Qbert einigermaßen sicher über die Würfel hüpfen läßt, kommt bald angenehm ins Schwitzen, denn es geht sehr schnell, sehr turbulent und auch farbenfroh zu. Das Spielhallengame wurde nahezu perfekt für den TI adaptiert, der Spielablauf mag vielleicht nicht ganz so umfangreich sein wie bei den großen Maschinen, aber ich hatte auch mit der TI-Version großen Spaß. Selbstverständlich existieren mehrere Spielrunden, die sich durch wachsende Geschwindigkeit auszeichnen.

Wenn die Konzentration auch nur etwas nachläßt, hat man schnell seine Qberts verloren und muß wieder von vorne beginnen. Wir wollen hoffen, daß sich noch mehr Hersteller bemühen, solch gelungene Spiele für den TI auf den Markt zu bringen. Qbert jedenfalls zeigt mit hervorragender grafischer Gestaltung und eindrucksvoller Soundunterstützung, zu welchen Leistungen der alte TI imstande ist. Da stellt er manche Mitkonkurrenten in den Schatten.

Klaus Weidemann

	Note		Note
Grafik:	❷	**Action:**	❶
Sound:	❷	**Spielwitz:**	❷
Gesamtergebnis:		**Spielidee:**	❸

This collection is for all TI99/4a user.
If at you like TI99 computer join at international forum present on AtariAge.com Site.
Greetings from Italy. Astharot

Der Wüstenplanet